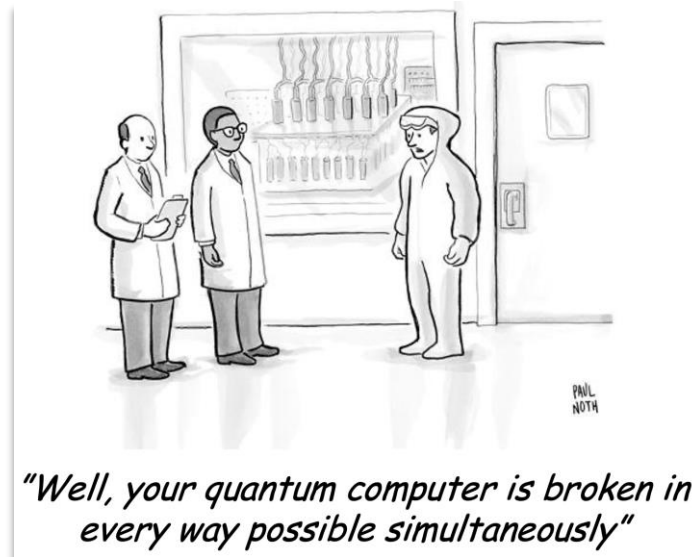


Reinforcement learning in CERN's accelerators and beyond

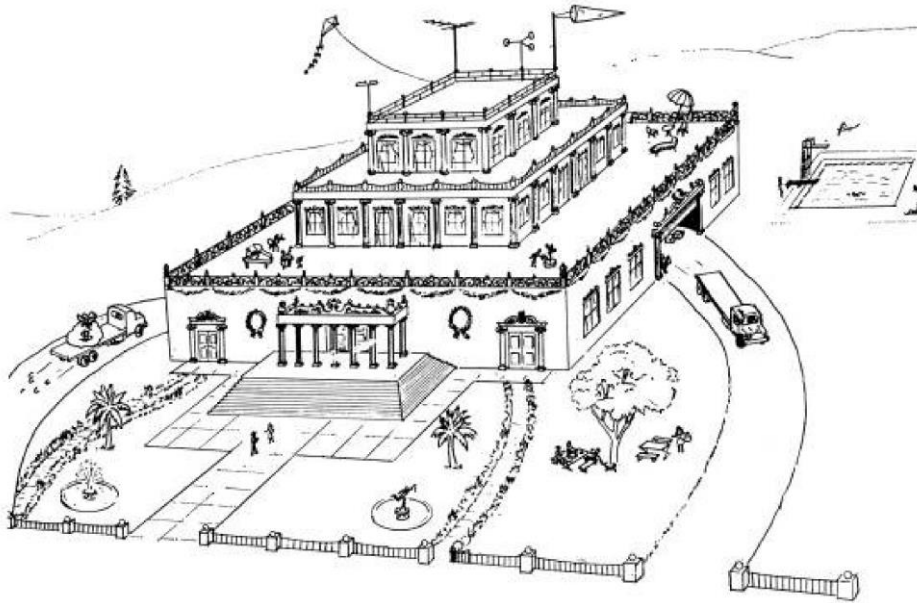


M. Schenk, M. Grossi, V. Kain, K. Li, S. Vallecorsa
CERN, Switzerland

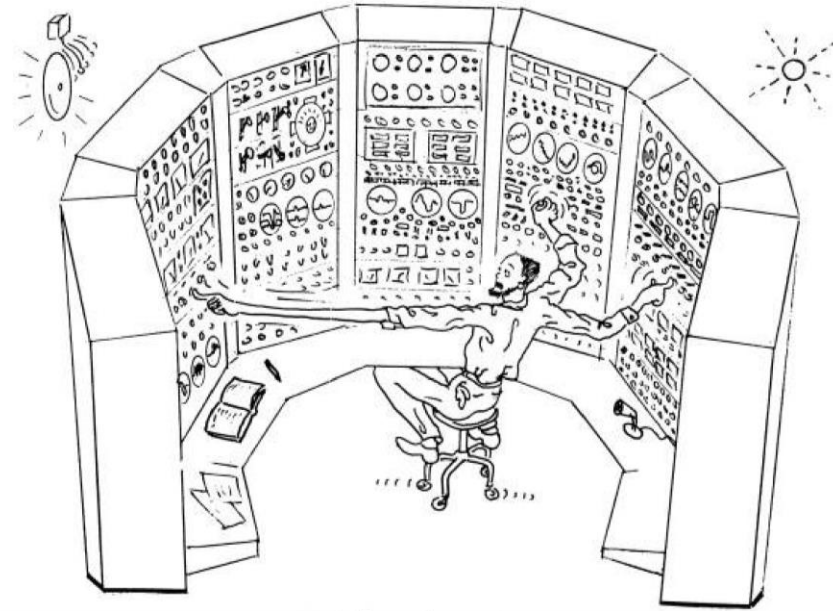
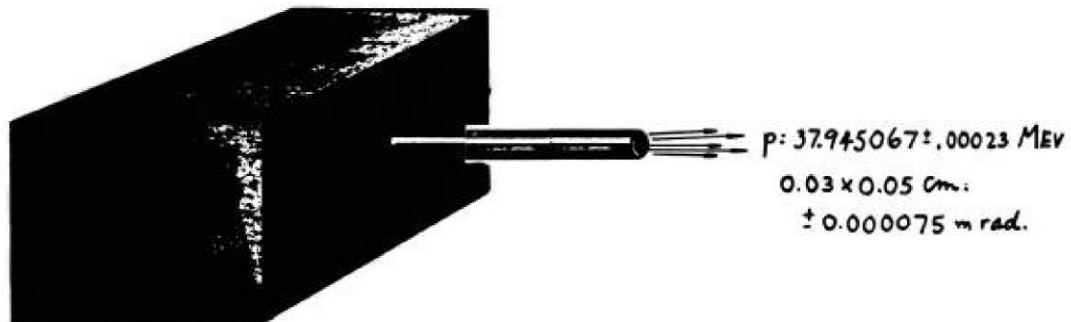
E. F. Combarro
University of Oviedo, Spain

Accelerators seen by the ...

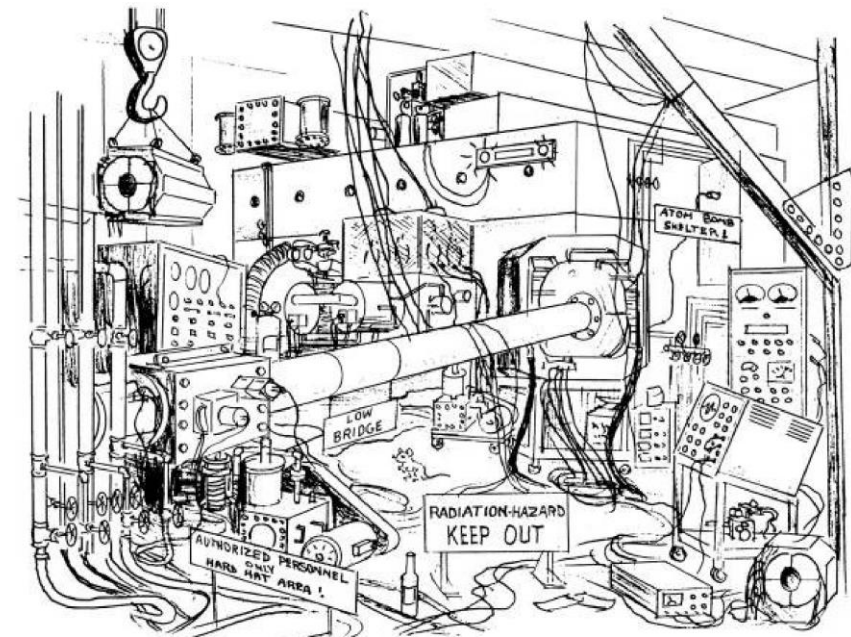
... funding agency



... experimental physicist



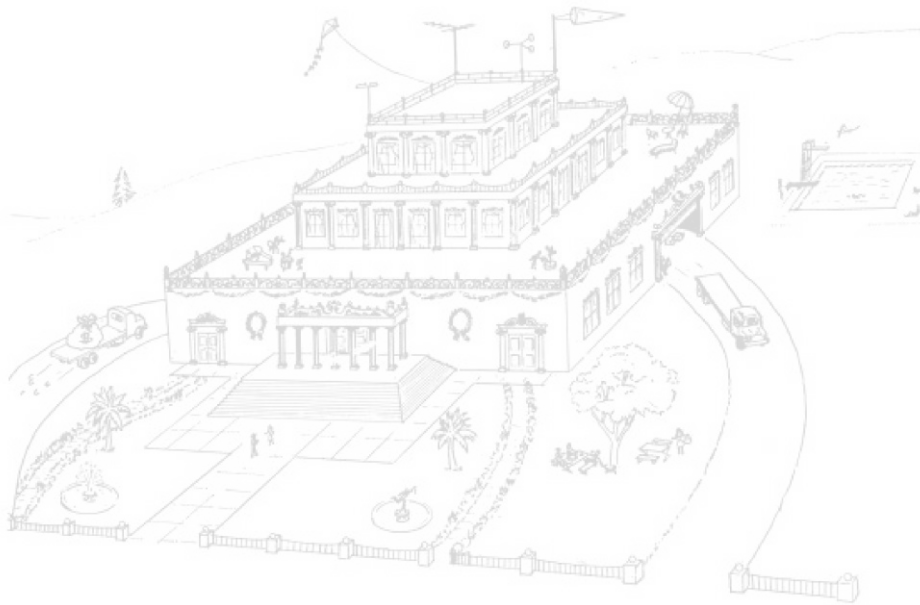
... machine operators & beam physicists



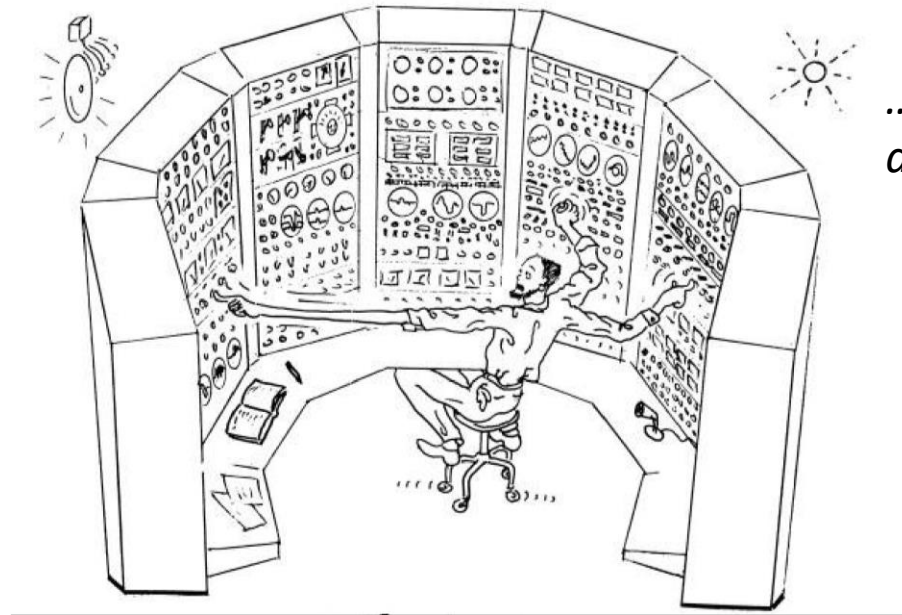
... visitors

Accelerators seen by the ...

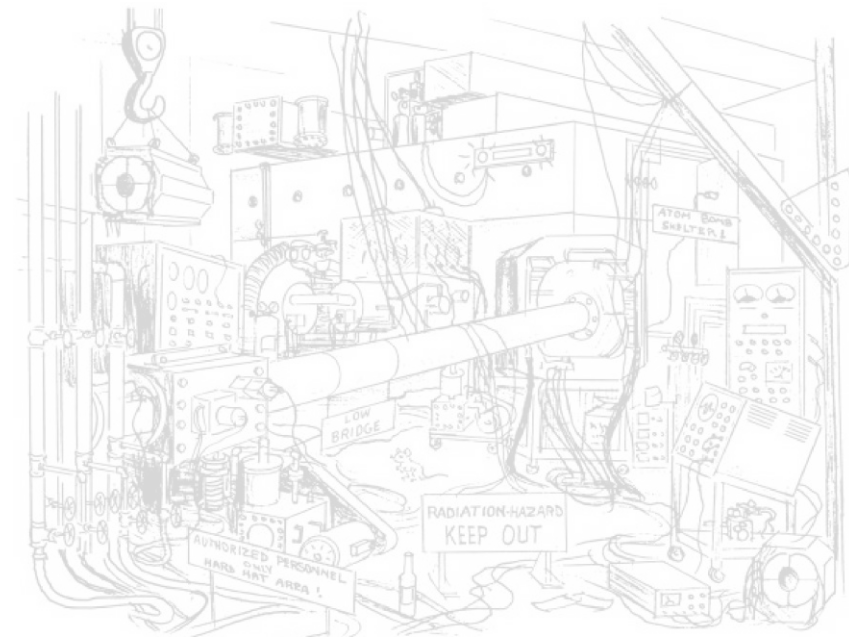
... funding agency



... experimental physicist



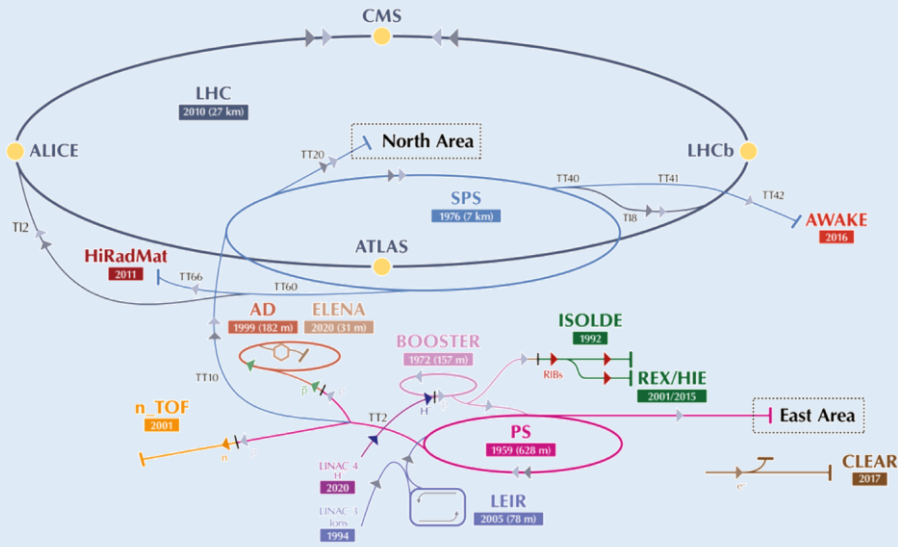
... machine operators & beam physicists



... visitors

Introduction

Motivation



CERN maintains **dense & diverse physics program**

➔ Imposes **challenges on accelerator operation**

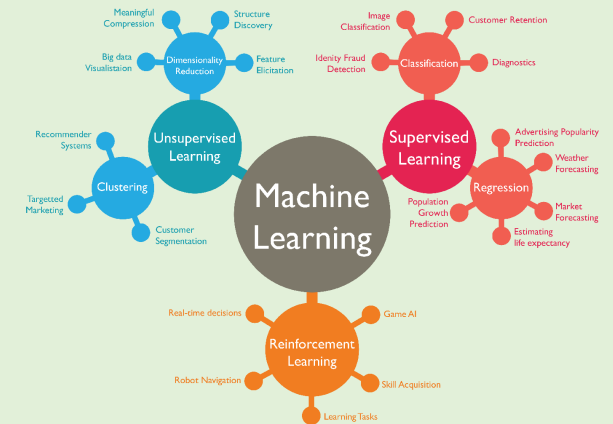
- **Machine availability & beam quality** critical for experiments to reach physics objectives
- Broad spectrum of machine & beam types requiring **high flexibility** and **efficient parameter tuning**

Ideally, machine operation **based on physics models**

Not always possible: complexity, completeness, online evaluation, ...

➔ **Explore new technologies**

- What does the **optimization & ML toolbox** have to offer?
- Will discuss **specific aspects of reinforcement learning (RL)** today



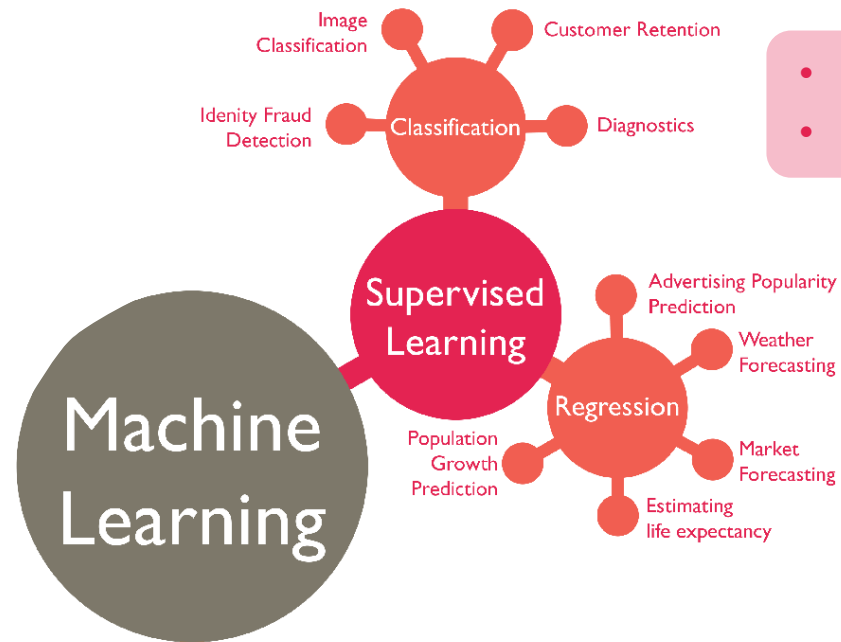
Introduction

RL and the machine learning landscape

- **Goal:** find structure in data
- **Data:** unlabelled samples (x_i)

Other approaches

- Classic control
- Numerical and Bayesian optimization, (GP-)MPC, fuzzy logic, etc.
- Some of them very suitable for accelerator-related problems



- **Goal:** find mapping $f: x_i \mapsto y_i$
- **Data:** labelled samples (x_i, y_i)

- **Goal:** learn to take optimal decisions
- **Data:** agent actively interacts with environment collecting samples & rewards

[*Image source*](#)

Contents

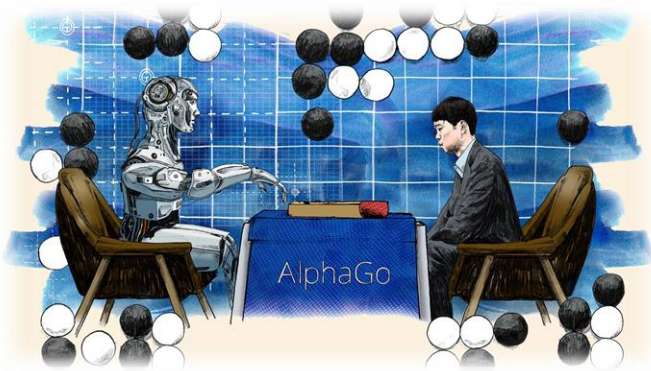
- **Introduction**
- **Reinforcement learning**
- **Beyond classical RL**
- **Results**
- **Conclusions & outlook**

Contents

- Introduction
- **Reinforcement learning**
- Beyond classical RL
- Results
- Conclusions & outlook

Reinforcement learning (RL)

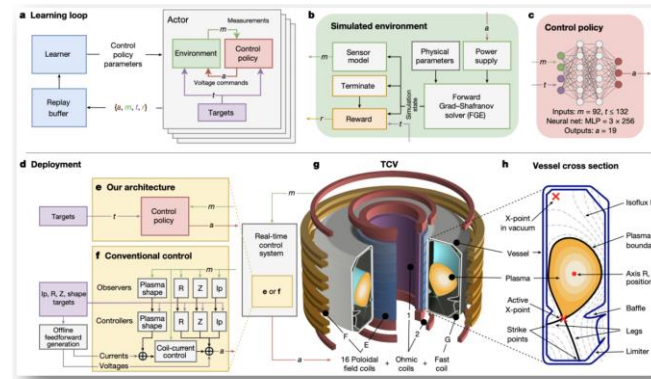
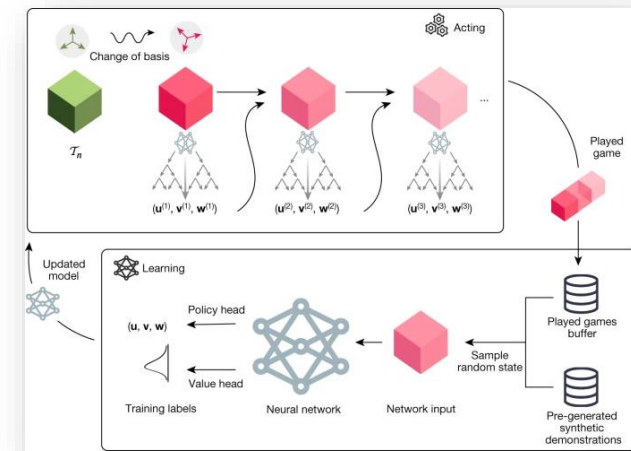
State-of-the-art



DeepMind, 2016: [AlphaGo](#)

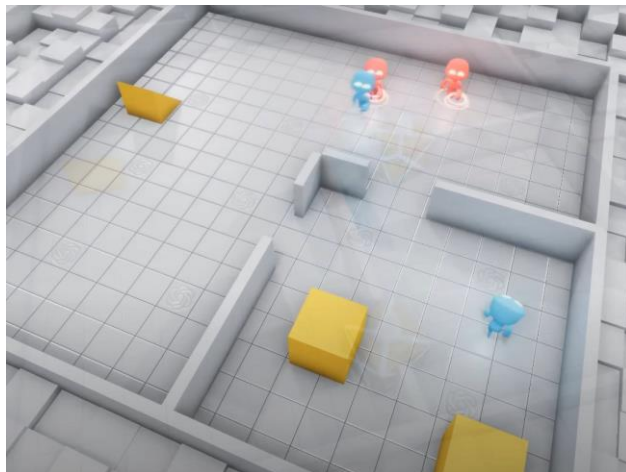
DeepMind, 2022: [AlphaTensor](#)

- Improve computational efficiency of matrix multiplication
- Benefits countless fields
- RL agent discovered more efficient algorithms than developed by humans



DeepMind & EPFL, 2022: [Tokamak control](#)

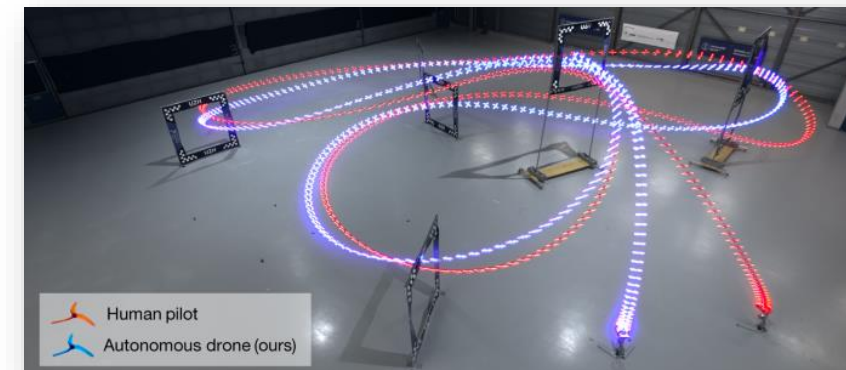
- Maintaining plasma within tokamak requires high-dimensional, high-frequency, closed-loop control using magnetic actuator coils
- RL agent successfully trained as magnetic controller



OpenAI, 2019: [Hide and seek](#)

UZH & Intel Labs, 2023: [Drone racing](#)

- RL agent beats human drone racing champions in real environment
- Training in simulations with mixed-in residual models from real data



and more ...

Reinforcement learning (RL)

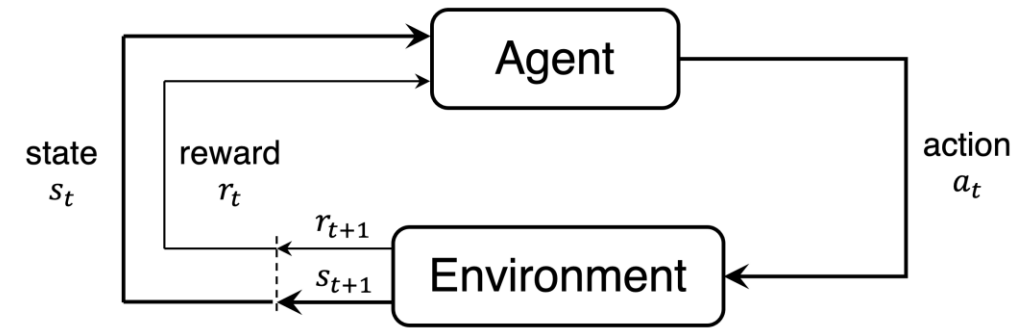
... in a nutshell

Trial-and-error learning

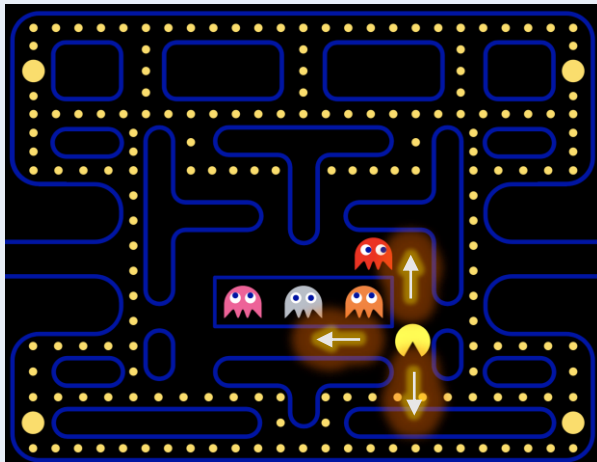
- Agent **takes actions** in environment and **collects rewards**
- **Goal**
 - learn optimal behavior \Leftrightarrow **maximize return G_t**
 - sum of discounted, future rewards

$$G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k} \quad \text{with } \gamma \in (0, 1)$$

- **Policy π** : encodes agent's behavior – “what action to take in a given state?”
- **Can be solved in various ways**: many kinds of algorithms exist



Sutton & Barto



Example: Pacman

State: where am I? Where are ghosts, snacks, cookies?

Actions: up, down, left, right

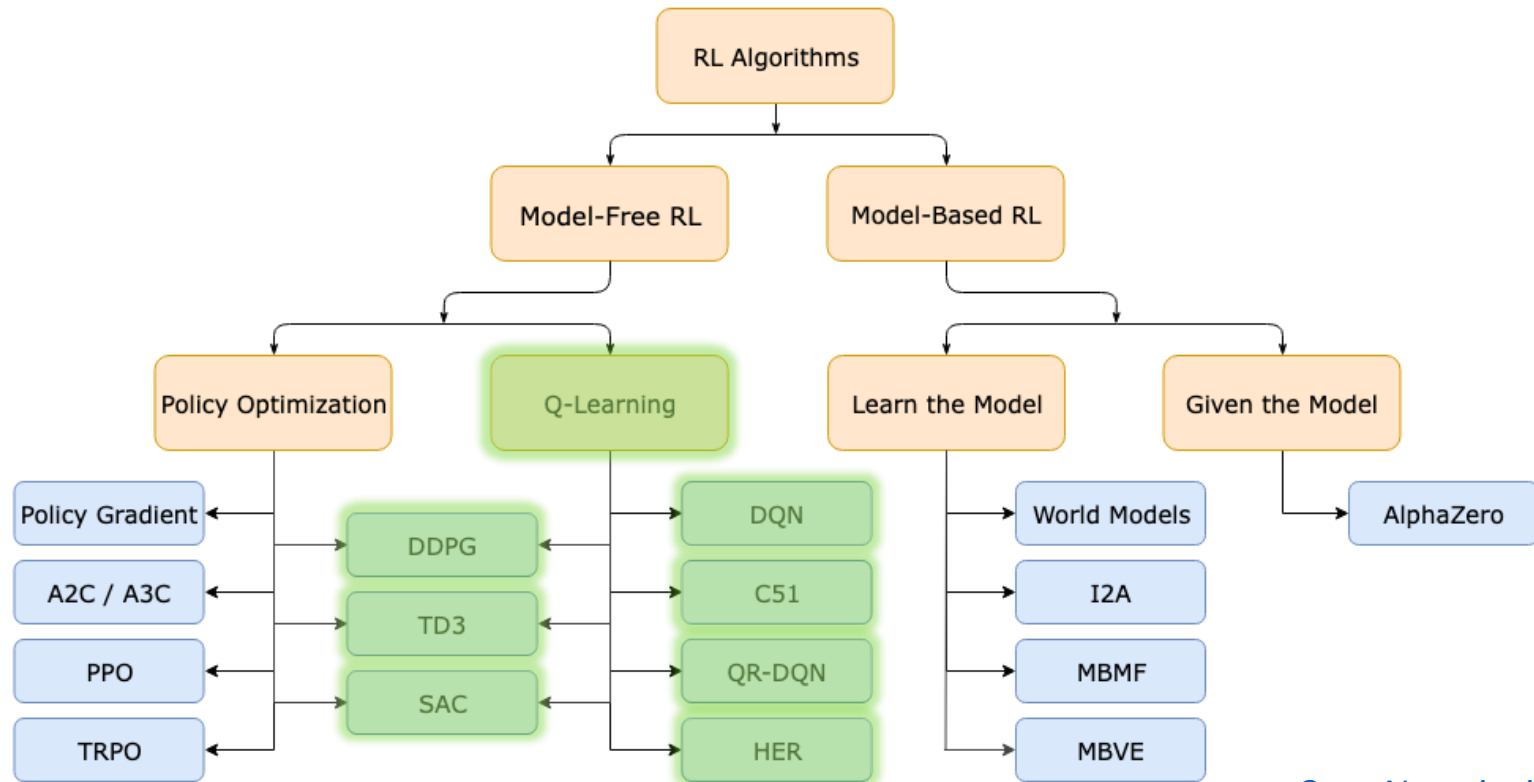
Reward: food (+), ghosts (-)

Return: how much food am I going to eat over time

Policy: given state, should I go up, down, left, or right?

Reinforcement learning (RL)

Algorithm zoo



[OpenAI – spinning up](#)

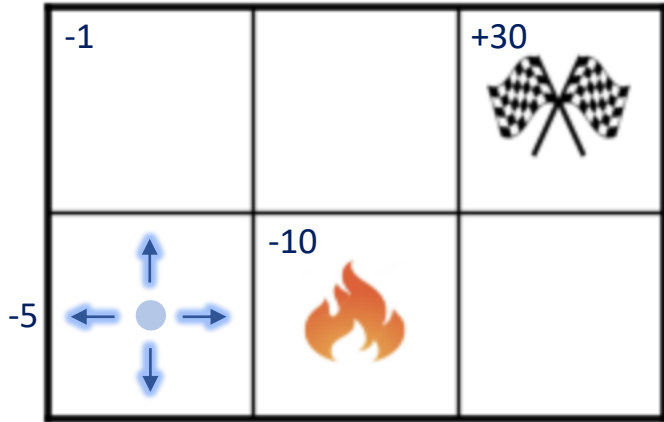
Focus on Q-learning

- Can understand a **large selection of RL algorithms**, and this talk
- Also covers algorithms **mostly employed for CERN RL applications** so far

Q-learning

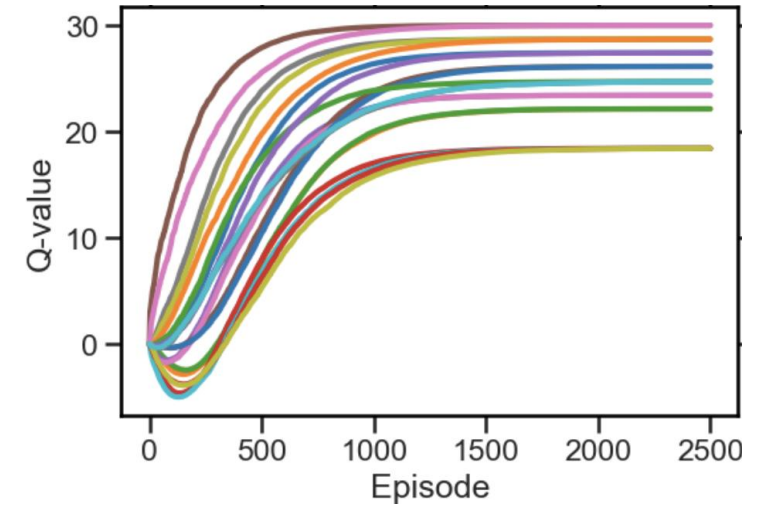
An example: Q-table

N.B.: we often deal with (large) continuous state-action spaces

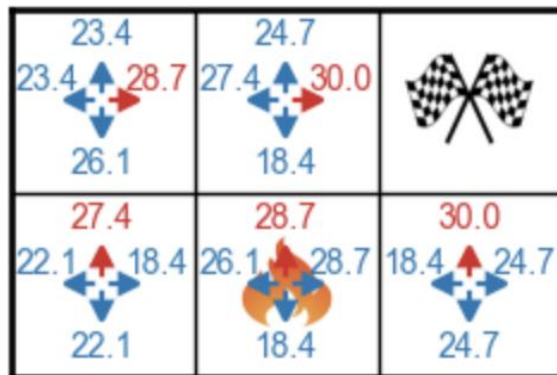


Episode 2400

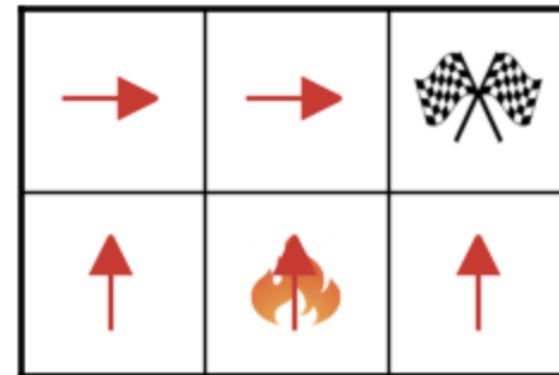
s \ a	up	down	left	right
(0, 0)	27.4	22.1	22.1	18.4
(0, 1)	23.4	26.1	23.4	28.7
(1, 0)	28.7	18.4	26.1	28.7
(1, 1)	24.7	18.4	27.4	30.0
(2, 0)	30.0	24.7	18.4	24.7
(2, 1)	0.0	0.0	0.0	0.0



$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha [r_t + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)]$$



greedy policy

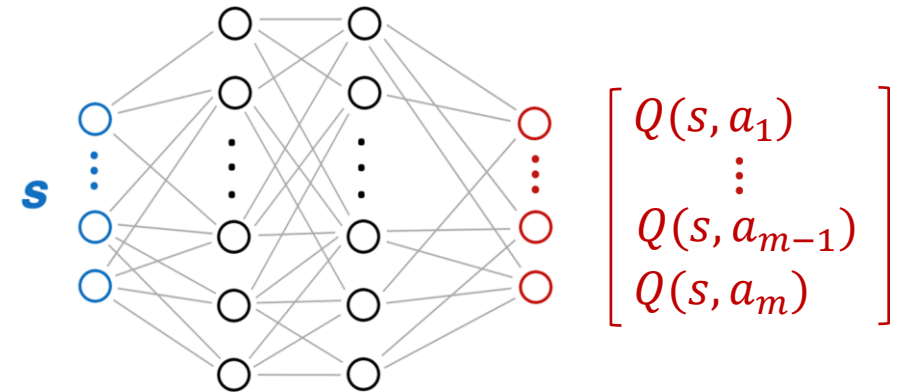
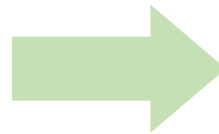


Q-learning

Deep Q-learning (DQN)

Episode 2400

s \ a	up	down	left	right
(0, 0)	27.4	22.1	22.1	18.4
(0, 1)	23.4	26.1	23.4	28.7
(1, 0)	28.7	18.4	26.1	28.7
(1, 1)	24.7	18.4	27.4	30.0
(2, 0)	30.0	24.7	18.4	24.7
(2, 1)	0.0	0.0	0.0	0.0

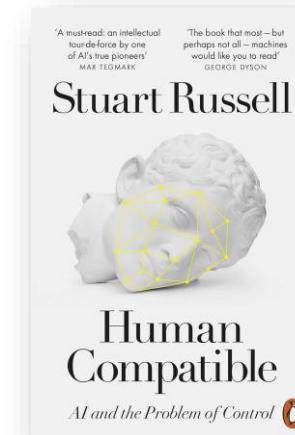


- Q-function of **continuous or very large state-spaces** can no longer be represented by a table
- **Replace table by a neural network:** deep Q-learning (DQN)
 - universal function approximator
 - great interpolator (e.g. for unseen states)
- Train network weights using **temporal difference learning**
- Does not yet solve the **continuous action problem**: see e.g. actor-critic schemes

RL challenges

... for accelerator applications, but not only

- **Sample efficiency**
 - How many agent-environment interactions are required for convergence?
 - Relevant for **particle accelerator control** given cost of beam time (*online training*)
- **Reward engineering**
 - Alignment: getting the objective right
“Making sure the agent does what we want it to do”
- **State definition**
 - Ideally “fully observable”
 - Limited beam instrumentation
 - Can be a real RL show-stopper
- **Parameter drifts**
 - Dependencies we can not easily include in our state
- **Safety**
 - Particularly a concern during exploration
 - There are ways to add safety to RL agents



arXiv > cs > arXiv:2205.10330

Computer Science > Artificial Intelligence

[Submitted on 20 May 2022 (v1), last revised 20 Feb 2023 (this version, v4)]

A Review of Safe Reinforcement Learning: Methods, Theory and Applications

Shangding Gu, Long Yang, Yali Du, Guang Chen, Florian Walter, Jun Wang, Yaodong Yang, Alois Knoll

Reinforcement learning (RL) has achieved tremendous success in many complex decision making tasks. When it comes to deploying RL in

<https://arxiv.org/abs/2205.10330>

RL challenges

... for accelerator applications, but not only

- **Sample efficiency**

- **How many agent-environment interactions** are required for convergence?
- Relevant for **particle accelerator control** given cost of beam time (*online training*)



- **Reward engineering**

- Alignment: getting the objective right
“Making sure the agent does what we want it to do”

- **State definition**

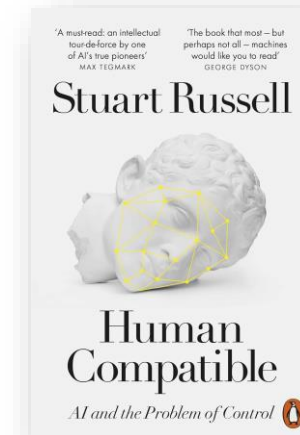
- Ideally “fully observable”
- Limited beam instrumentation
- Can be a real RL show-stopper

- **Parameter drifts**

- Dependencies we can not easily include in our state

- **Safety**

- Particularly a concern during exploration
- There are ways to add safety to RL agents



arXiv > cs > arXiv:2205.10330

Computer Science > Artificial Intelligence

[Submitted on 20 May 2022 (v1), last revised 20 Feb 2023 (this version, v4)]

A Review of Safe Reinforcement Learning: Methods, Theory and Applications

Shangding Gu, Long Yang, Yali Du, Guang Chen, Florian Walter, Jun Wang, Yaodong Yang, Alois Knoll

Reinforcement learning (RL) has achieved tremendous success in many complex decision making tasks. When it comes to deploying RL in

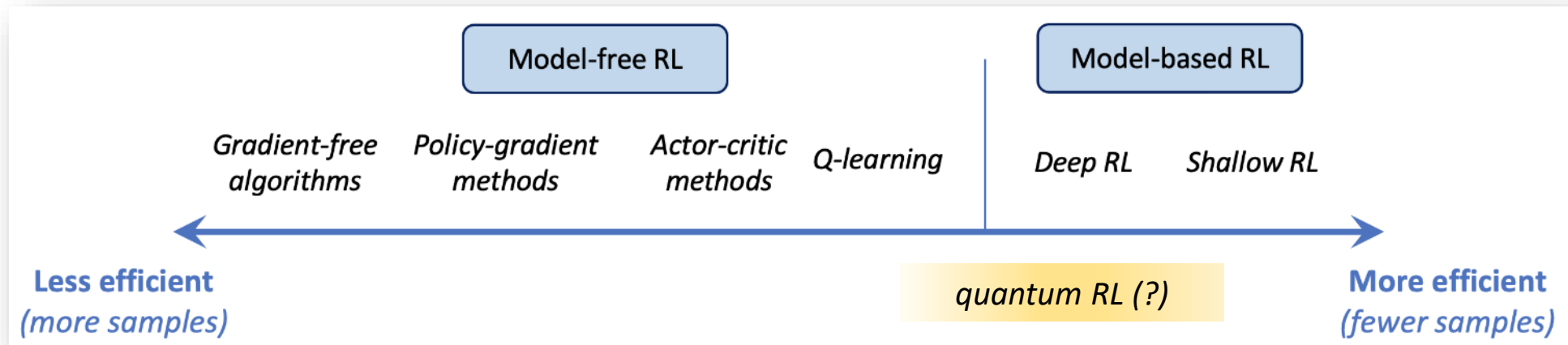
<https://arxiv.org/abs/2205.10330>

RL challenges

Sample efficiency

How to improve sample efficiency?

- **Reliable simulations / surrogate models**
Train RL agent on model, then deploy in real world (*sim2real*)
- **Choice of algorithm**
 - **Model-based RL:** learn model explicitly and train agent on that (= *planning*)
 - **Quantum RL?**



Contents

- Introduction
- Reinforcement learning
- **Beyond classical RL**
- Results
- Conclusions & outlook

Beyond classical RL

Free energy based approach

Q-learning performance depends on type of function approximator we use for $Q(s, a)$

- **Classical deep Q-learning (DQN)**
Feed-forward neural net
- **Free-energy based RL (FERL)**
Quantum Boltzmann machine (QBM)

Free energy-based reinforcement learning using a quantum processor

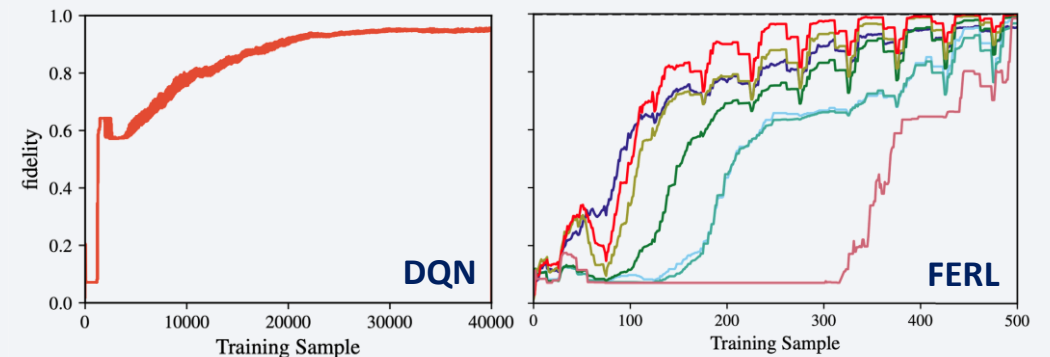
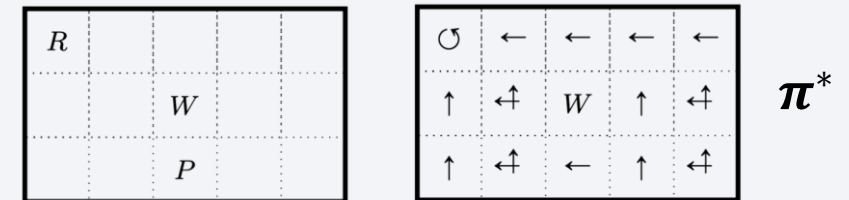
Anna Levit,¹ Daniel Crawford,¹ Navid Ghadermarzy,^{1,2} Jaspreet S. Oberoi,^{1,3} Ehsan Zahedinejad,¹ and Pooya Ronagh^{1,2,*}

¹QBit, 458-550 Burrard Street, Vancouver (BC), Canada V6C 2B5

²Department of Mathematics, The University of British Columbia, 121-1984 Mathematics Road, Vancouver (BC), Canada V6T 1Z2

³School of Engineering Science, Simon Fraser University, 8888 University Drive, Burnaby (BC), Canada V5A 1S6

Recent theoretical and experimental results suggest the possibility of using current and near-future quantum hardware in challenging sampling tasks. In this paper, we introduce free energy-based reinforcement learning (FERL) as an application of quantum hardware. We propose a method for processing a quantum annealer's measured qubit spin configurations in approximating the free energy of a quantum Boltzmann machine (QBM). We then apply this method to perform reinforcement learning on the grid-world problem using the D-Wave 2000Q quantum annealer. The experimental results show that our technique is a promising method for harnessing the power of quantum sampling in reinforcement learning tasks.



— D-Wave $\Gamma = 0.5, \beta = 2.0$ — SQA Chimera $\Gamma = 0.5, \beta = 2.0$
— D-Wave Classical $\beta = 2.0$ — SQA Bipartite $\Gamma = 0.5, \beta = 2.0$
— SA Chimera $\beta = 2.0$ — RBM
— SA Bipartite $\beta = 2.0$

<https://arxiv.org/pdf/1706.00074.pdf>

Beyond classical RL

Motivation: 1st study for 1D beam steering

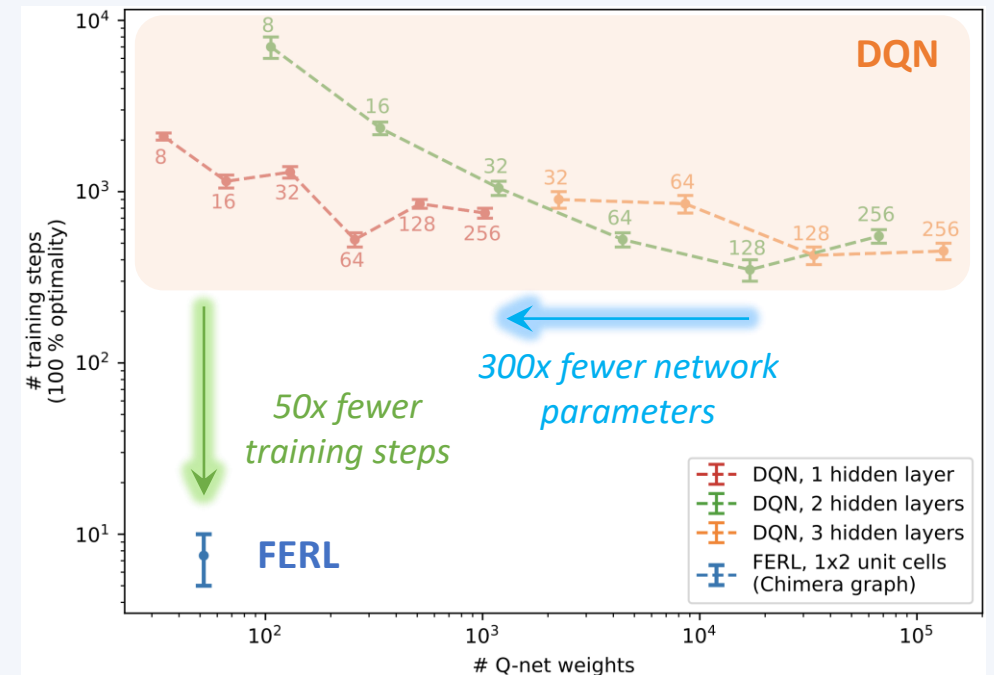
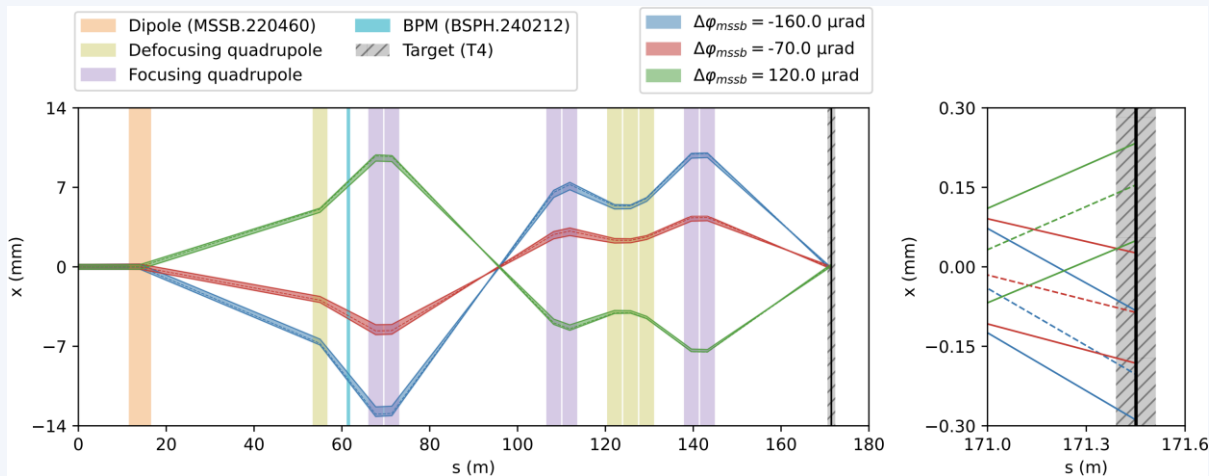
Our first attempt: 1D beam steering problem for SPS NA

- **Continuous state:** beam position (1D)
- **Discrete action:** $\pm\Delta$ on magnet kick angle (1D)
- **Reward:** amount of beam on target

- This looks interesting
- How does FERL work?
Quantum Boltzmann machine combined with quantum annealing
- Can we test it on other accelerator physics problems?

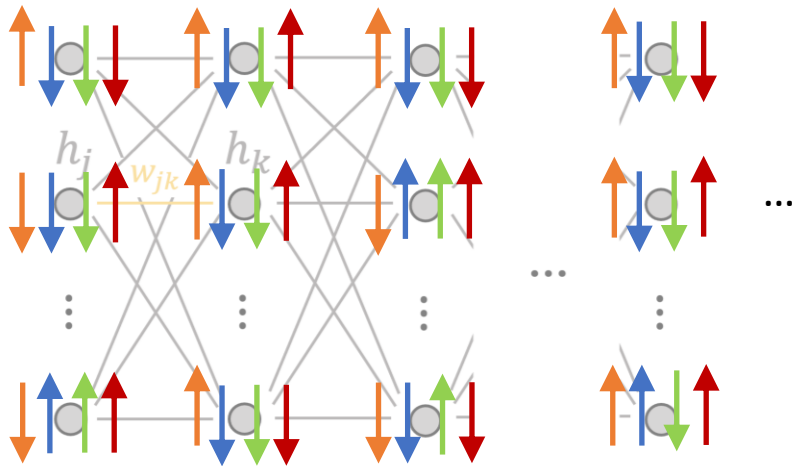
1D beam steering

North Area transfer line



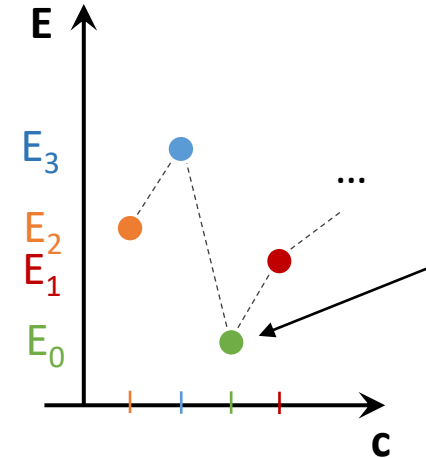
Beyond classical RL

Quantum Boltzmann machine



Hamiltonian of Ising spin model

$$\hat{\mathcal{H}} = -\frac{1}{2} \sum_{i,j} w_{ij} s_i^z s_j^z - \Gamma \sum_i s_i^x$$



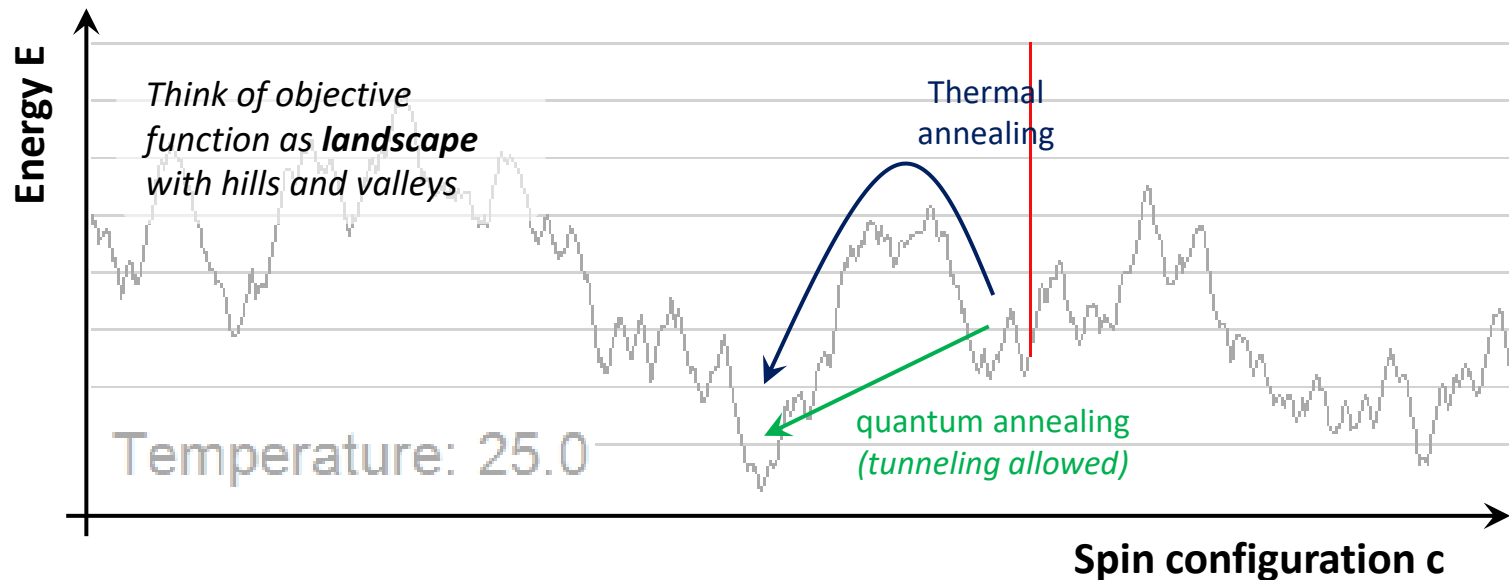
Ground state energy is what we care about, our **Q-value approximator**:
 $Q(s, a) \approx E_0$

- Network of **stochastic, binary** (*spin up / down*) **nodes**: coupled qubits
- **Every time we measure** qubit states, they assume **values according to probability distribution** over spin states
- Different spin configurations correspond to the system's **different energy levels** (2^N states)
- **Qubits influence** each other through **coupling weights** w_{jk} : this is how we “train” the QBM
- How to ensure system reaches **ground state** E_0 ? ➔ (simulated) **quantum annealing**

Beyond classical RL

Annealing: looking for the system's ground state

- **Simulated annealing**
 - Method for **function optimization** inspired by metallurgy: heating followed by **controlled cooling** for metals ➔ **strong crystalline structure**
 - **T high**: exploration
 - **$T \rightarrow 0$** : system assumes final configuration, ideally in global optimum
- **Quantum annealing**: higher efficiency thanks to quantum tunneling
- Way to bring **QBM into ground state** and **reliably get Q value** for RL



Quantum annealers

- Different type of quantum computer: non-gate based
- More qubits, but not as generic as gate-based QPUs
- Suited for quadratic minimization problems

Beyond classical RL

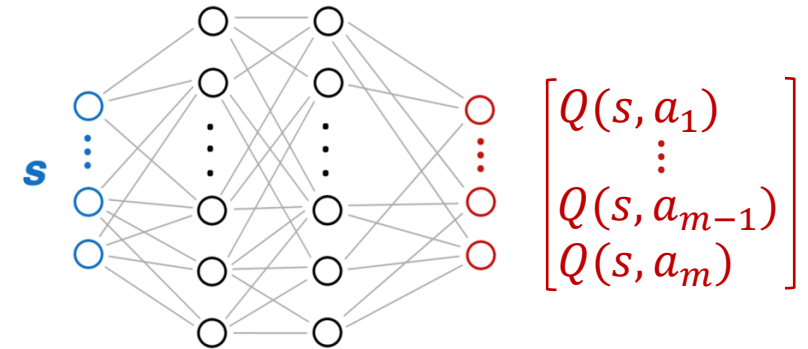
Comparison to deep Q-learning



Difference lies in how we approximate and predict Q

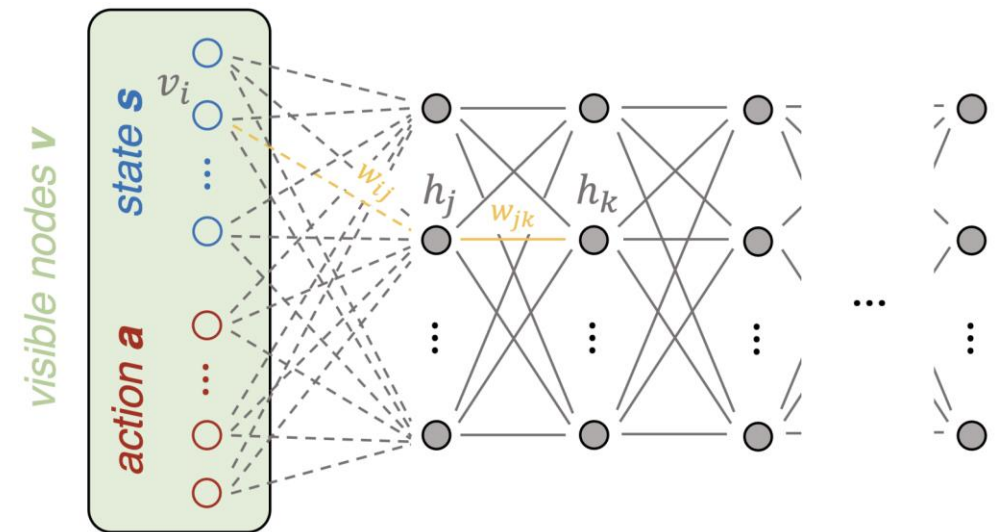
DQN

- **Network:** typically fully connected feed-forward
- Provide **state at input**
- **Q-value estimates** at output for **discrete set of actions**



FERL

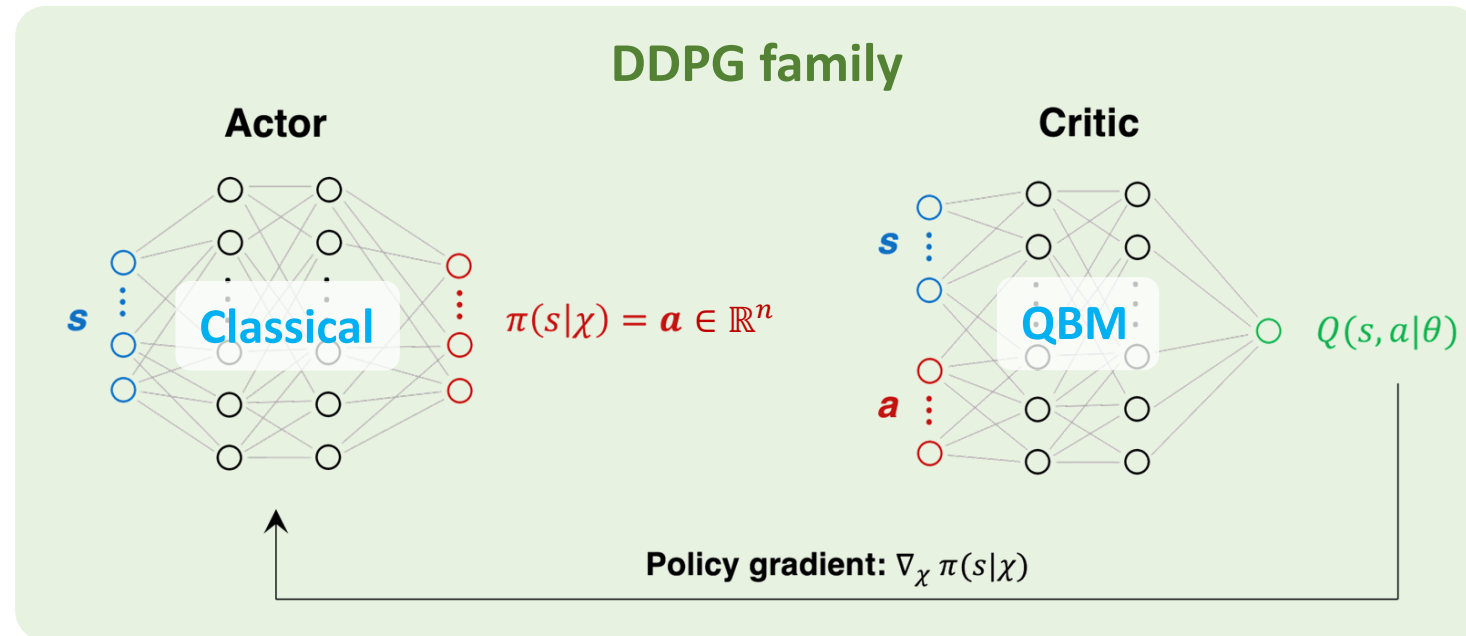
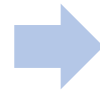
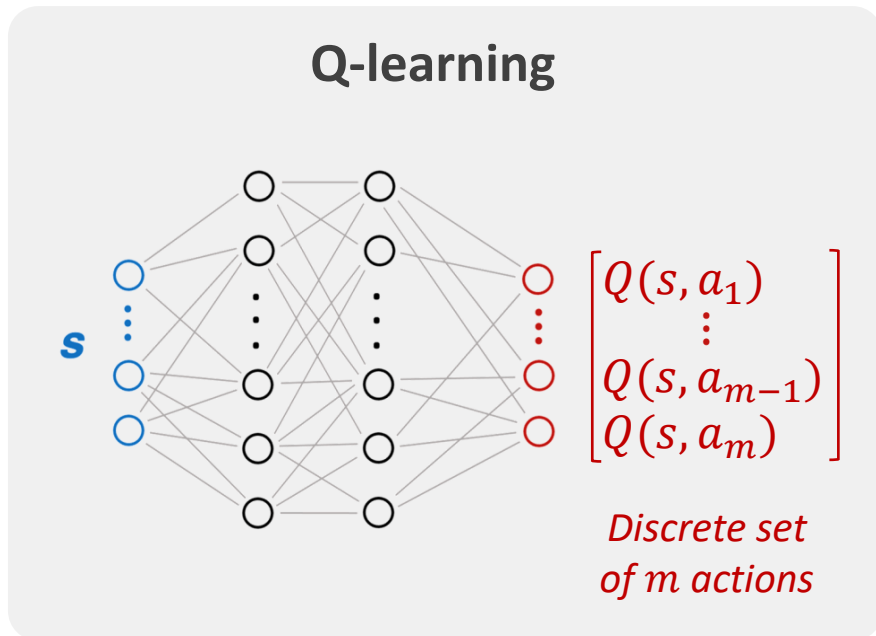
- **Network:** quantum Boltzmann machine
- **Implements energy model:** e.g. Ising spin model
- Provide **state and action at input**
- **Get $Q(s, a)$ through quantum annealing**
- **Why:** allows mapping to real physical system (= quantum annealer) with 1000s of qubits



Beyond classical RL

Developing a hybrid actor-critic scheme for continuous action-space problems

- Q-learning: **only discrete action-space** environments
- Accelerator optimization requires **continuous action space** → **develop hybrid actor-critic algorithm**
- Based on **Deep Deterministic Policy Gradient** family: DDPG, TD3, etc.
 - **QBM replaces classical critic, train with FERL approach**
 - Does it work? Can we exploit **high learning efficiency of FERL**?
Intuitively: if critic learns faster, should benefit actor training via policy gradient



Contents

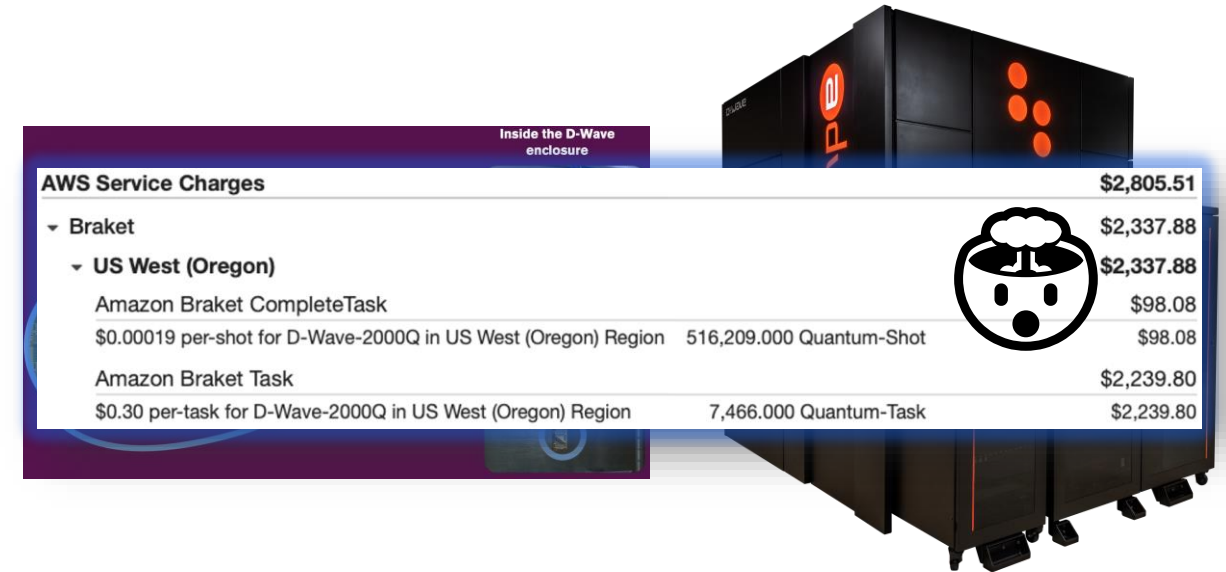
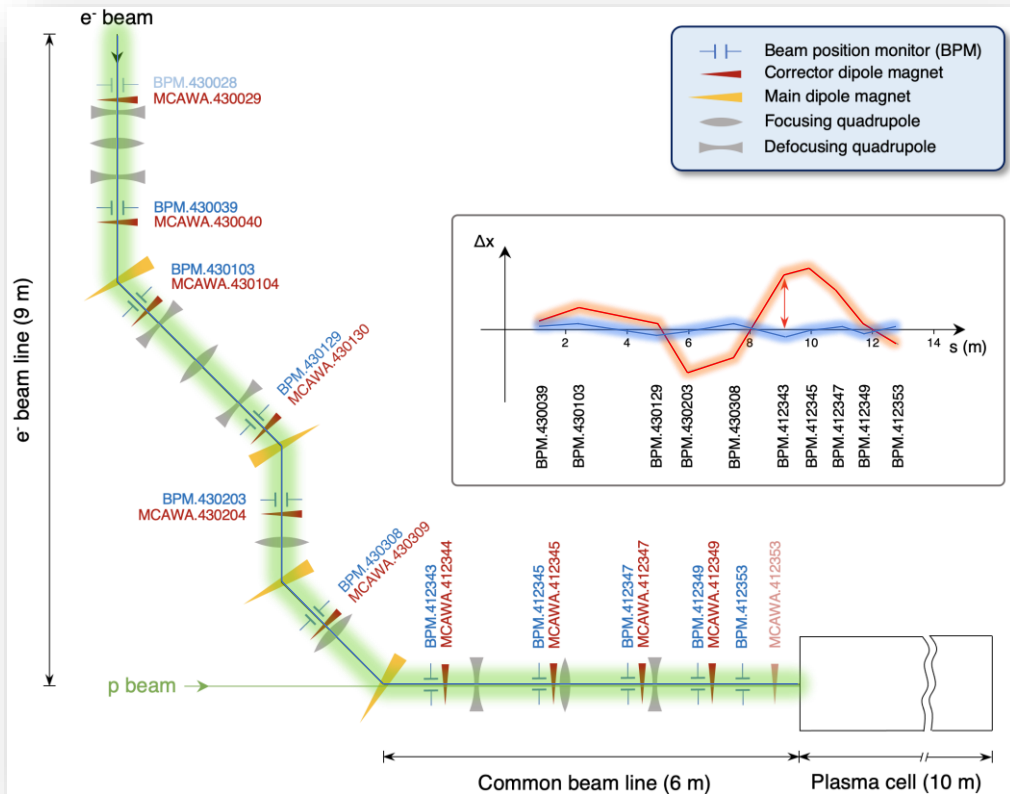
- Introduction
- Reinforcement learning
- Beyond classical RL
- **Results**
- Conclusions & outlook

2nd study: 10D continuous beam steering

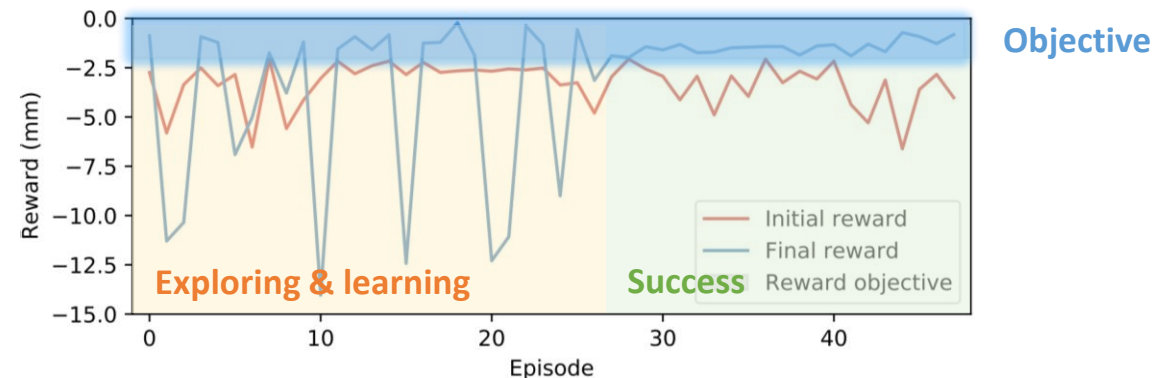
RL problem and training on quantum annealer

Framing the RL problem

- **Action:** deflection angles at 10 correctors
- **State:** beam positions at 10 BPMs
- **Objective:** minimize beam trajectory rms
 - ➔ **reward:** negative rms from 10 BPMs



Training: on D-Wave Advantage quantum annealer (QA)

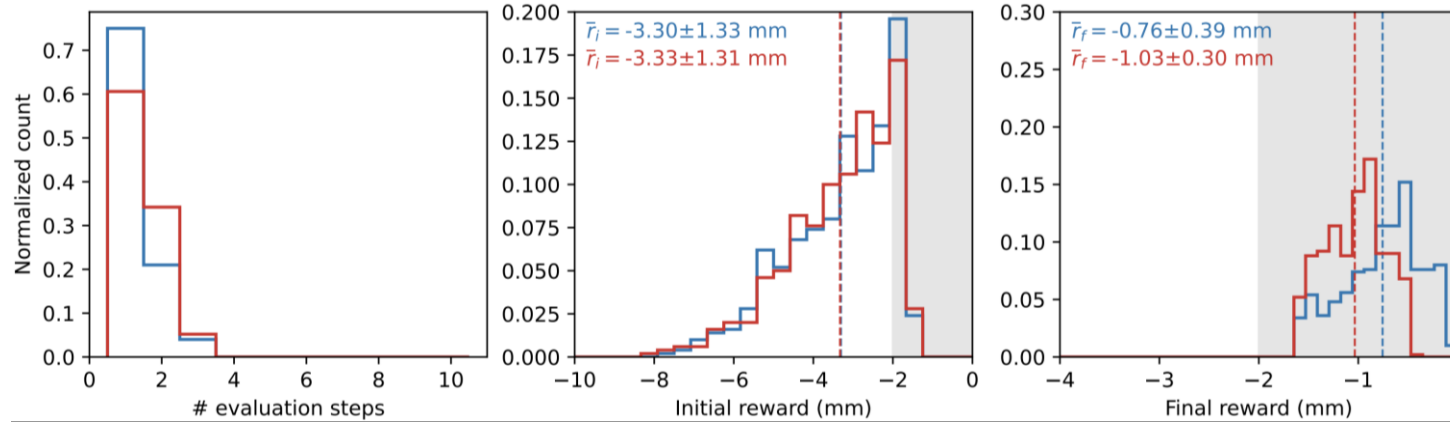


2nd study: 10D continuous beam steering

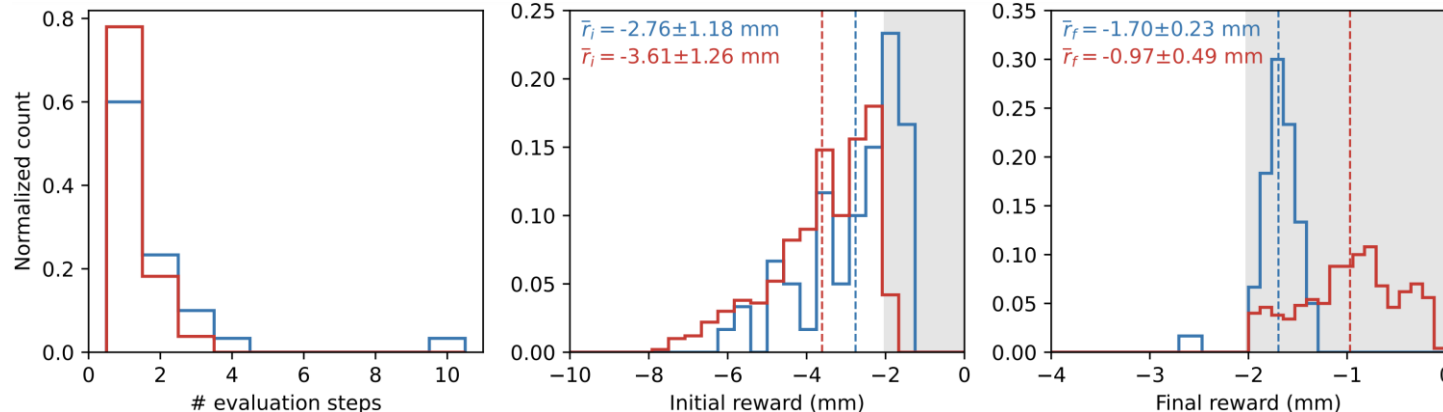
Evaluation

Real (QA) vs simulated (SQA) quantum annealing

Simulated beam line



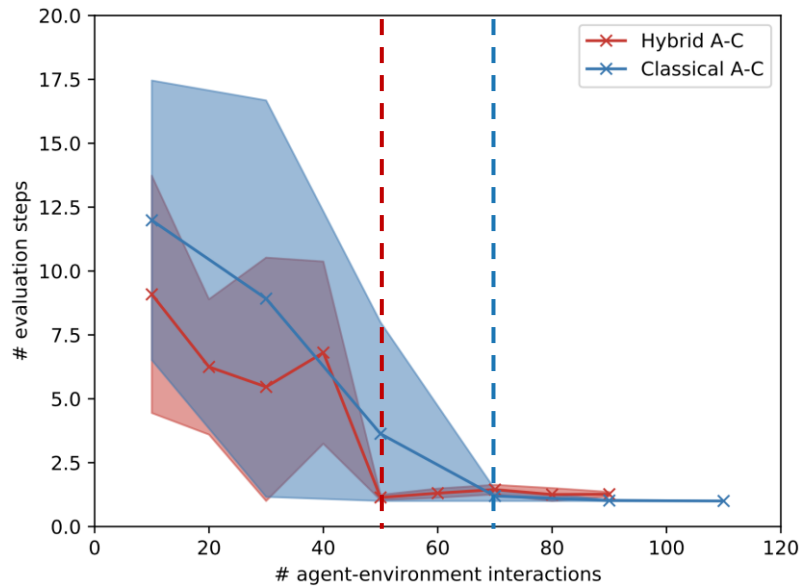
Real beam line



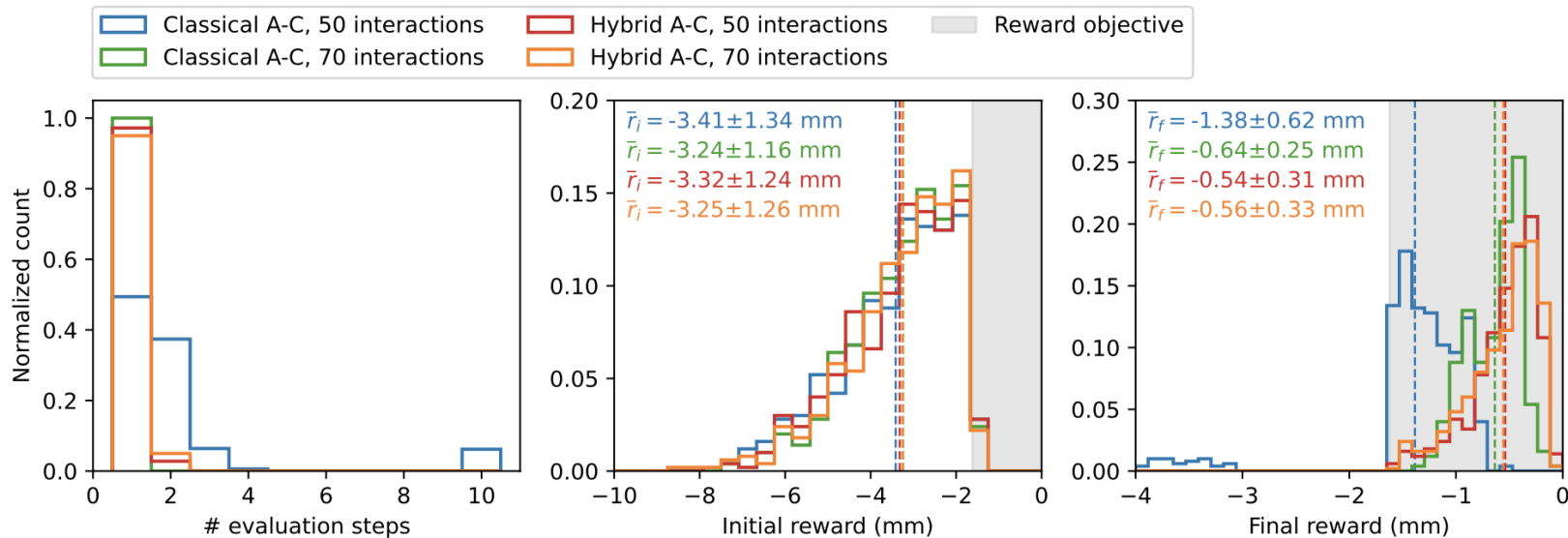
- But it works 😊 !
- Hyperparameter tuning with simulated quantum annealing
- Consistent performance on simulated line
- Agents minimize rms in **1 step** for **60 – 70 %** cases
- Both also perform fine on real beamline
- SQA-agent better on real line compared to QA agent

2nd study: 10D continuous beam steering

Comparison to a classical actor-critic



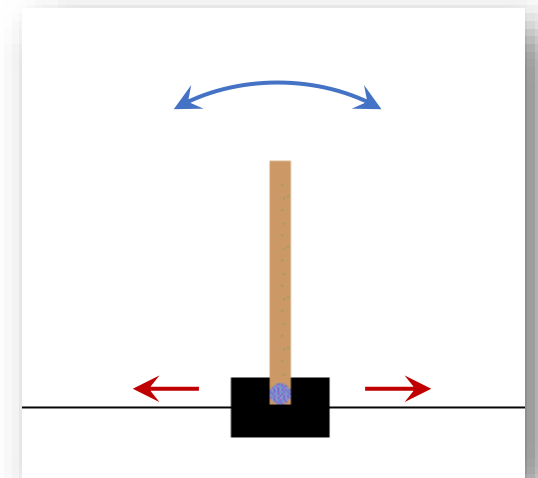
- **Slight improvement in terms of sample efficiency**
50 vs 70 interactions
- **Very few interactions sufficient** for both approaches
- **Dynamics potentially too simple** (linear problem)
➔ Move towards **non-linear RL benchmarks**



3rd study: Cart-Pole v1

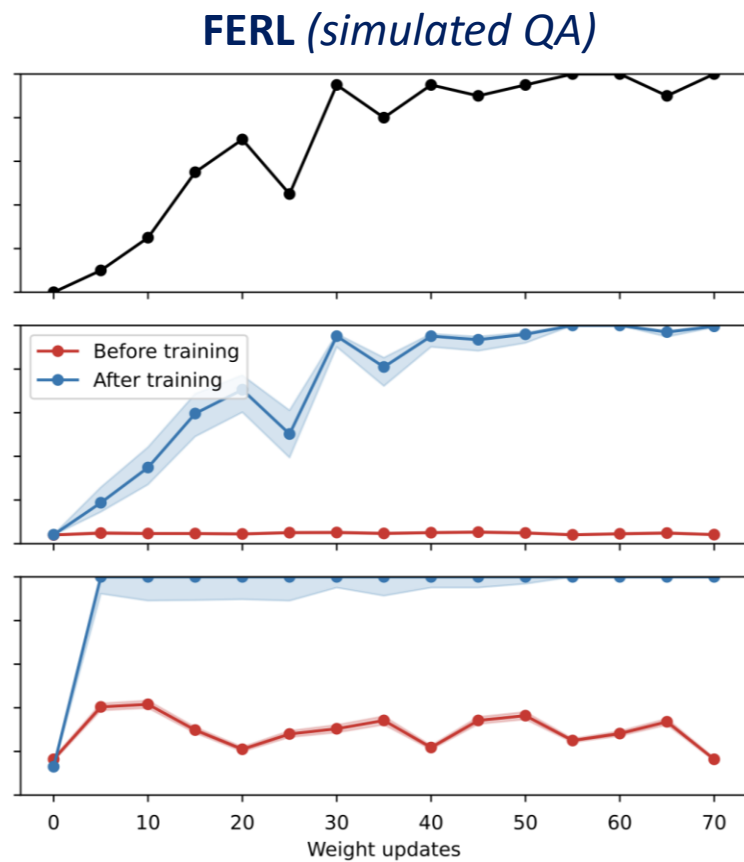
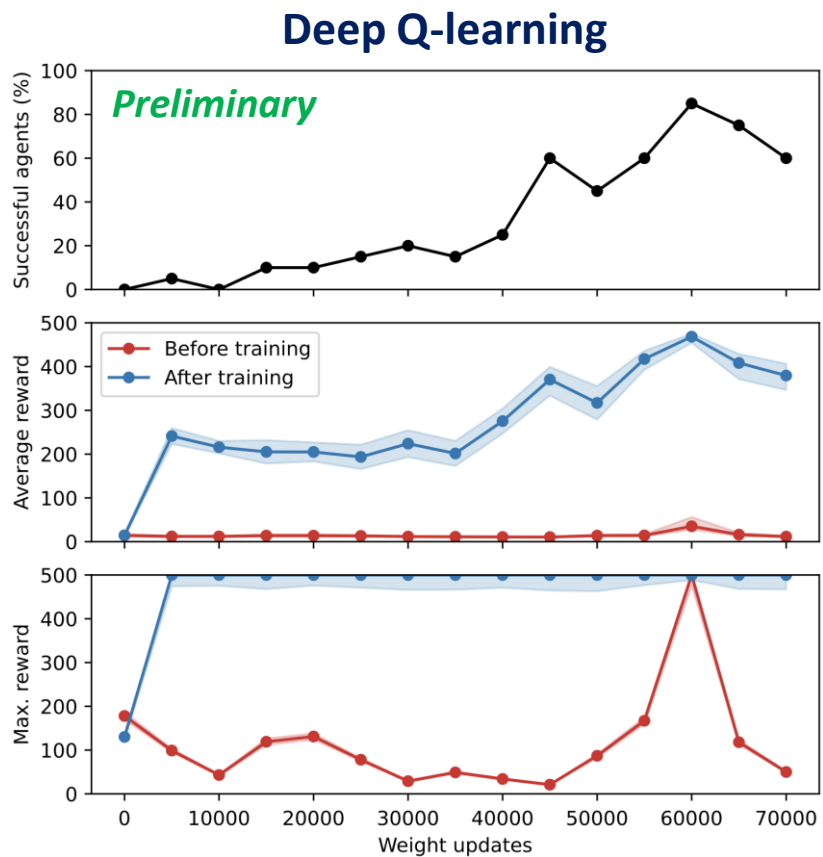
Overview

- **Cart-Pole v1:** official [env](#) from classic control problems domain
 - **Continuous state (4D):** cart position & velocity, pole angle & angular velocity
 - **Discrete action (1D):** push cart left or right
 - **Reward:** +1 per iteration
 - **Max. episode length:** 500 iterations
- **Non-linear dynamics**
- **Agents**
 - **Deep Q-learning:** various architectures optimized with ray-tune
 - **FERL:** 2x2 unit cell QBM (32 qubits), simulated QA and on D-Wave



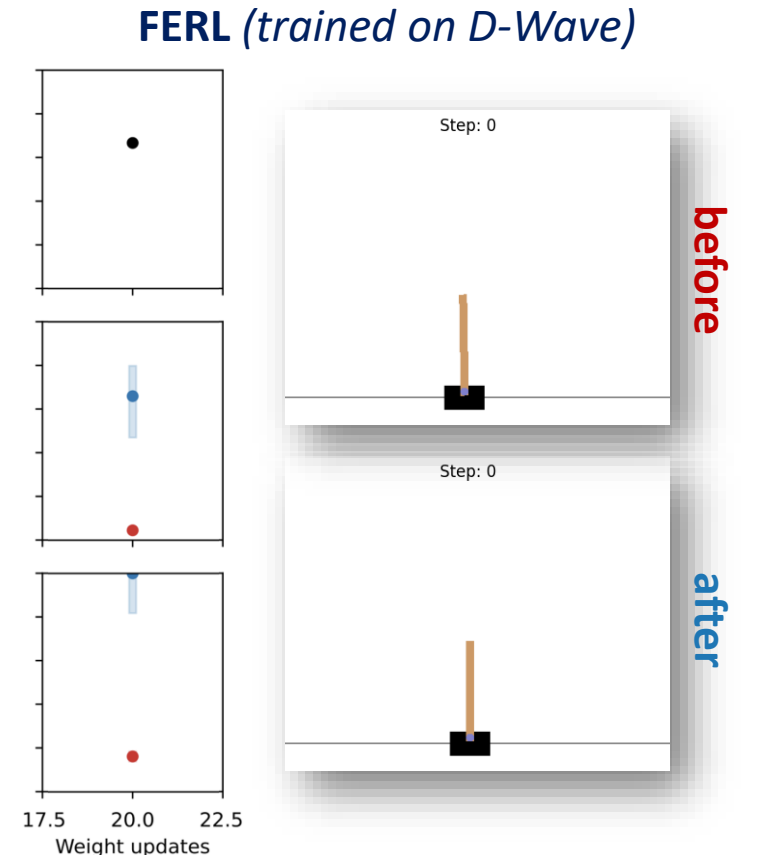
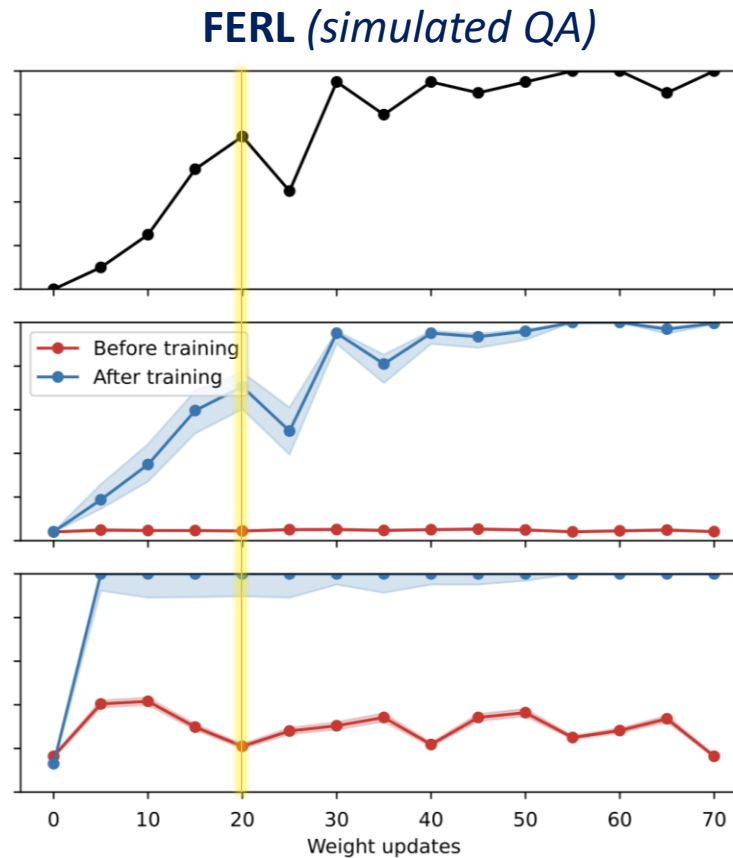
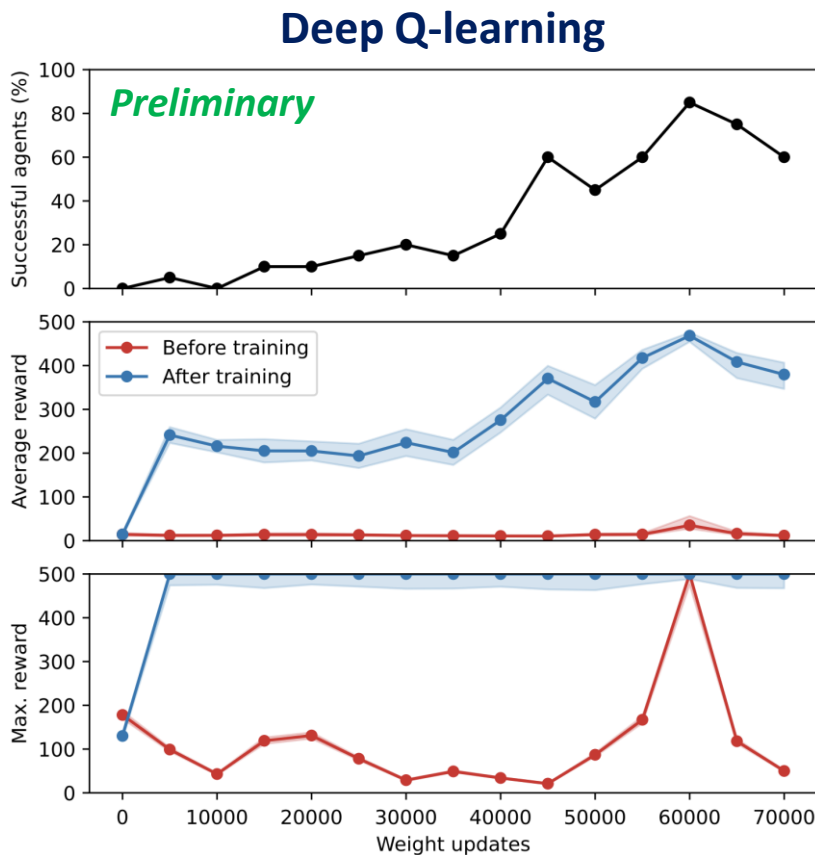
3rd study: Cart-Pole v1

Comparing DQN vs FERL



3rd study: Cart-Pole v1

Comparing with D-Wave hardware



- **Big gain in sample efficiency and robustness for FERL vs DQN**
- **Similar performance on D-Wave hardware**

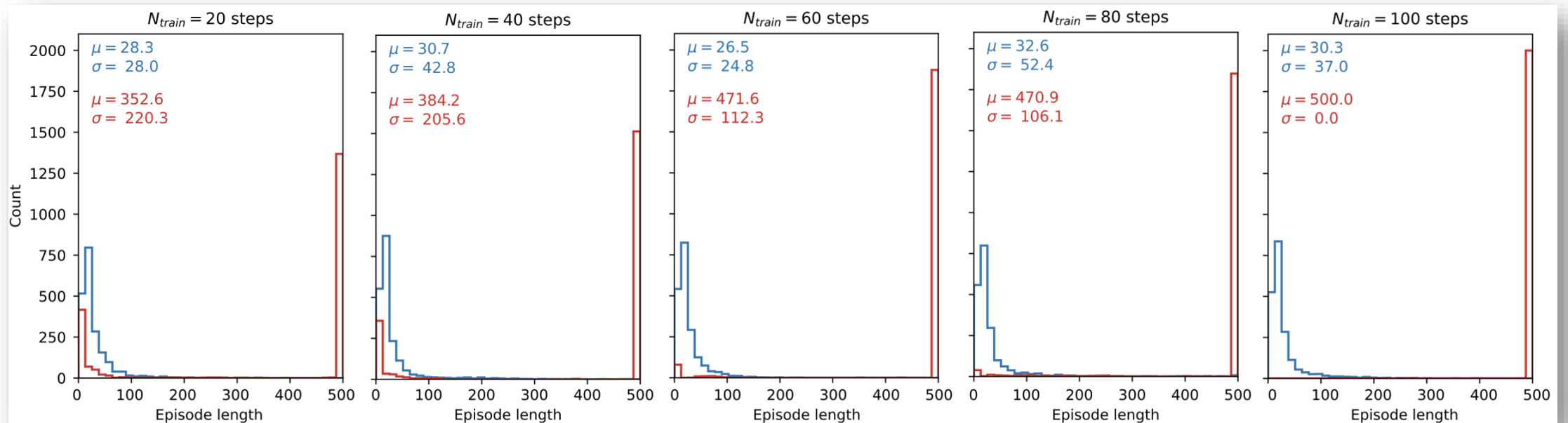
3rd study: Cart-Pole v1

More detailed study on FERL

- **Distribution of episode length** before and after different numbers of training iterations
 - 50 independent agents
 - 40 evaluation episodes
- **Same 40 initial states** for every agent
- $N_{\text{train}} = 100$: **all agents optimal**

Incredibly fast convergence

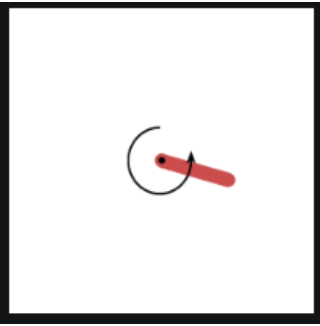
➔ **Next:** how about non-linear continuous action problems?



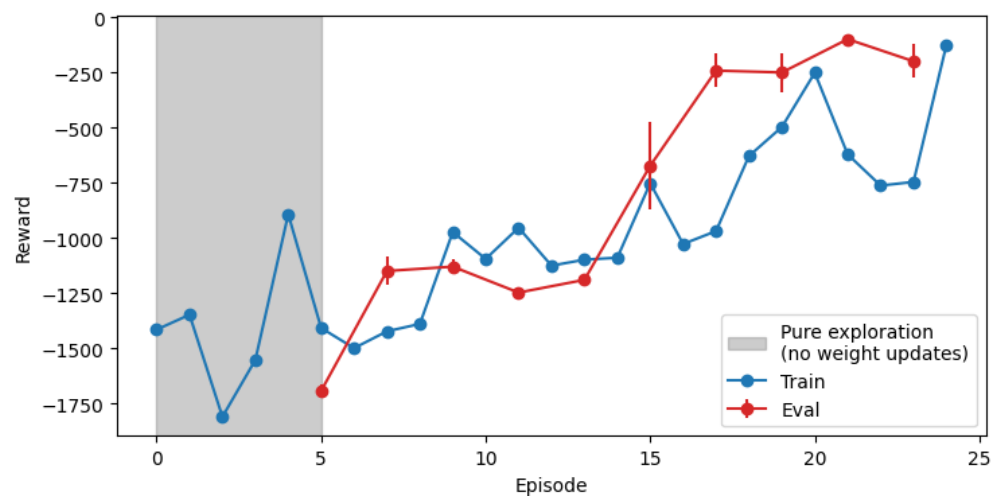
4th study: Pendulum-v1

Continuous action space

Action Space	Box(-2.0, 2.0, (1), float32)
Observation Shape	(3,)
Observation High	[1. 1. 8.]
Observation Low	[-1. -1. -8.]
Import	<code>gym.make("Pendulum-v1")</code>

A simple 2D diagram of a pendulum. A red rod is attached to a pivot point on the left, with a curved arrow indicating its angular displacement.

Classic DDPG (baseline)



Contents

- Introduction
- Reinforcement learning
- Beyond classical RL
- Results
- **Conclusions & outlook**

Some speculation

Why we think FERL could be more efficient than a classical approach?

Based on observations made ...

- **Thought 1**

- **QBM**s have **higher expressivity** compared to classical networks:
every node is defined by a probability distribution, rather than a simple matrix multiplication
- Hence require **fewer nodes than classical networks** to represent Q-function
- This means fewer parameters to fit and hence **less training data needed**
- **N.B.:** classical network with few nodes failed to solve the task

- **Thought 2**

- **QBM training itself is more efficient**, i.e. fewer weight updates needed to reach convergence
- Reason: **quantum annealing finds global optimum** more reliably

Final remarks & outlook

- **RL is powerful**, but comes with a number of **challenges**, such as achieving good **sample efficiency**
- **Depends on Q-approximator & algorithm:**
FERL shown to be **more sample efficient** than classical approach
- Developed **hybrid actor-critic** to solve **continuous action** problems
- **Successfully** trained on **simulated & real quantum annealer** and evaluated on **simulated & real AWAKE beam line**
- FERL on **non-linear Cart-Pole** problem shows very **fast convergence**
- **Outlook:** try different quantum hardware, progress on non-linear continuous action problem, explore alternative algorithm(s)



Hybrid actor-critic algorithm for quantum reinforcement learning at CERN beam lines

Michael Schenk¹, Elías F. Combarro², Michele Grossi¹, Verena Kain¹, Kevin Shing Bruce Li¹, Mircea-Marian Popa³ and Sofia Vallecorsa¹

¹European Organisation for Nuclear Research, Espl. des Particules 1, 1211 Meyrin, Switzerland

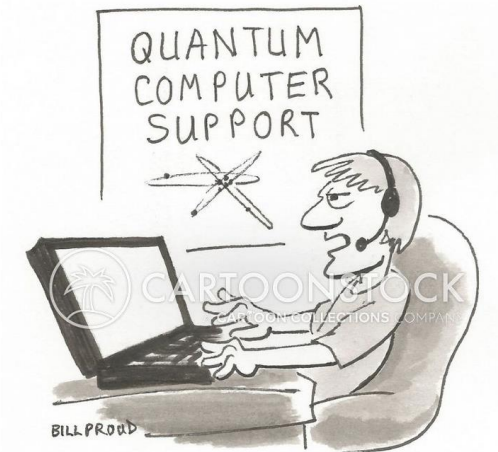
²Computer Science Department - University of Oviedo, C. San Francisco 3, 33003 Oviedo, Asturias, Spain

³Politehnica University of Bucharest, Splaiul Independentei 313, 060042 Bucharest, Romania

E-mail: michael.schenk@cern.ch

Abstract. Free energy-based reinforcement learning (FERL) with clamped quantum Boltzmann machines (QBM) was shown to significantly improve the learning efficiency

Questions?



"Have you tried turning it ON and OFF at the same time?"

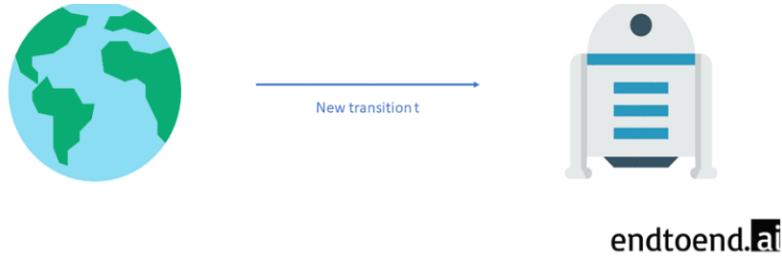
Backup

Beyond classical RL

Motivation: 1st study for 1D beam steering

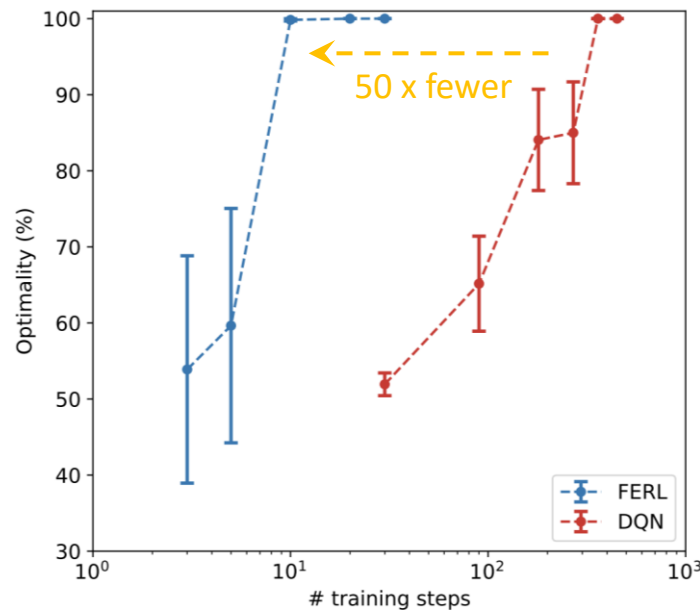
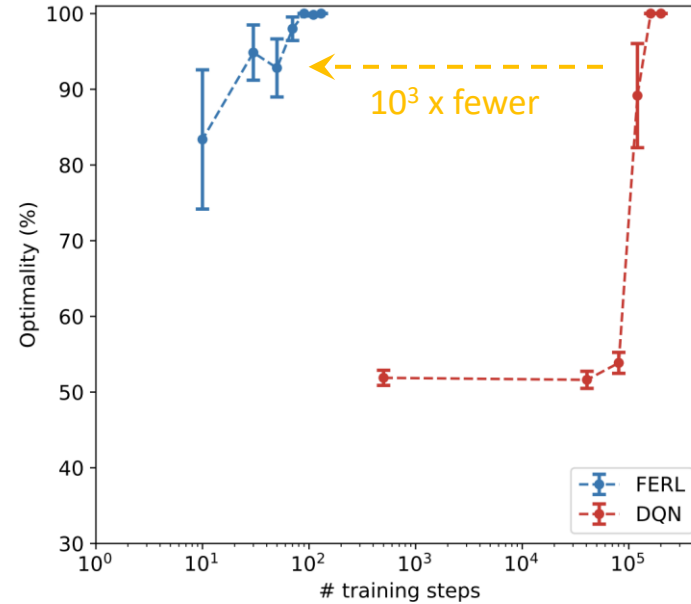
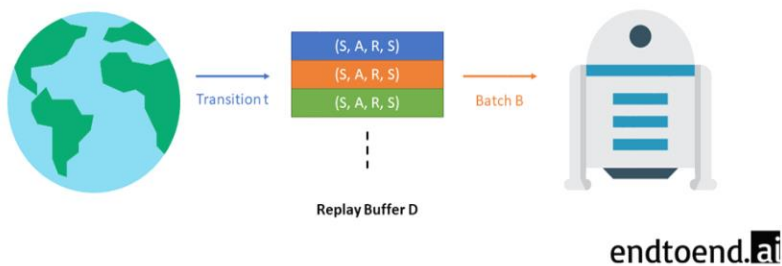
Online Learning

- Learn directly from latest experience
- Highly correlated data
- Agent learns from each interaction once and discards it immediately after



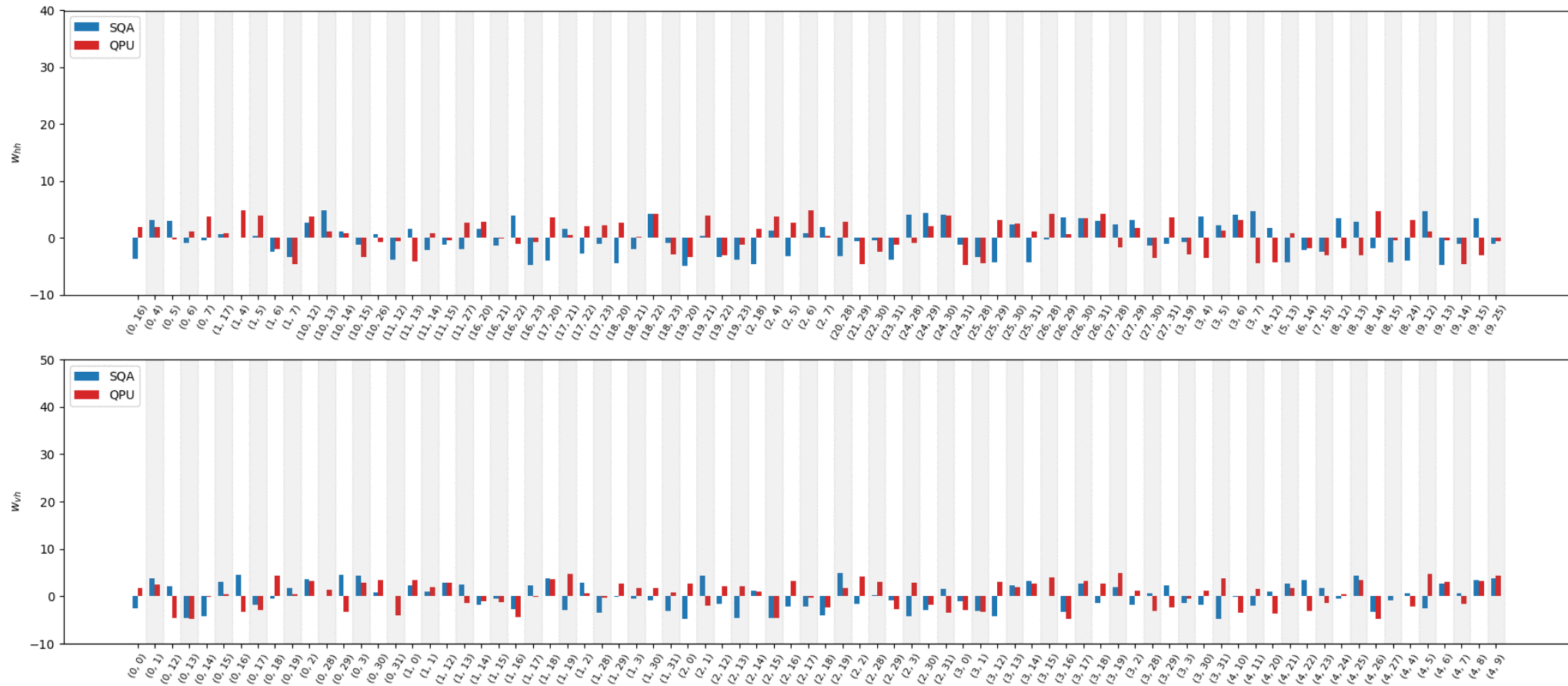
Experience Replay

- Save transitions into memory buffer
- Sample batch B from buffer to train agent at every step



3rd study: Cart-Pole v1

Coupling weights evolution: simulated vs real QA

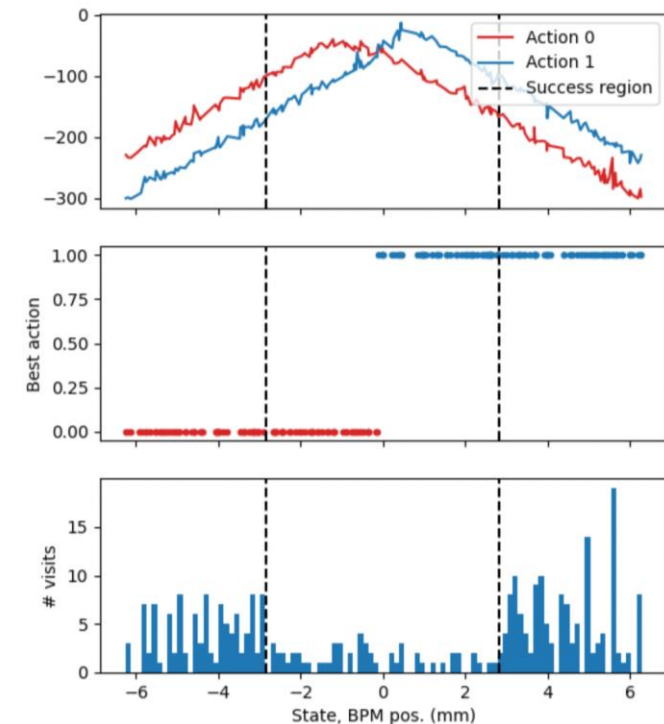
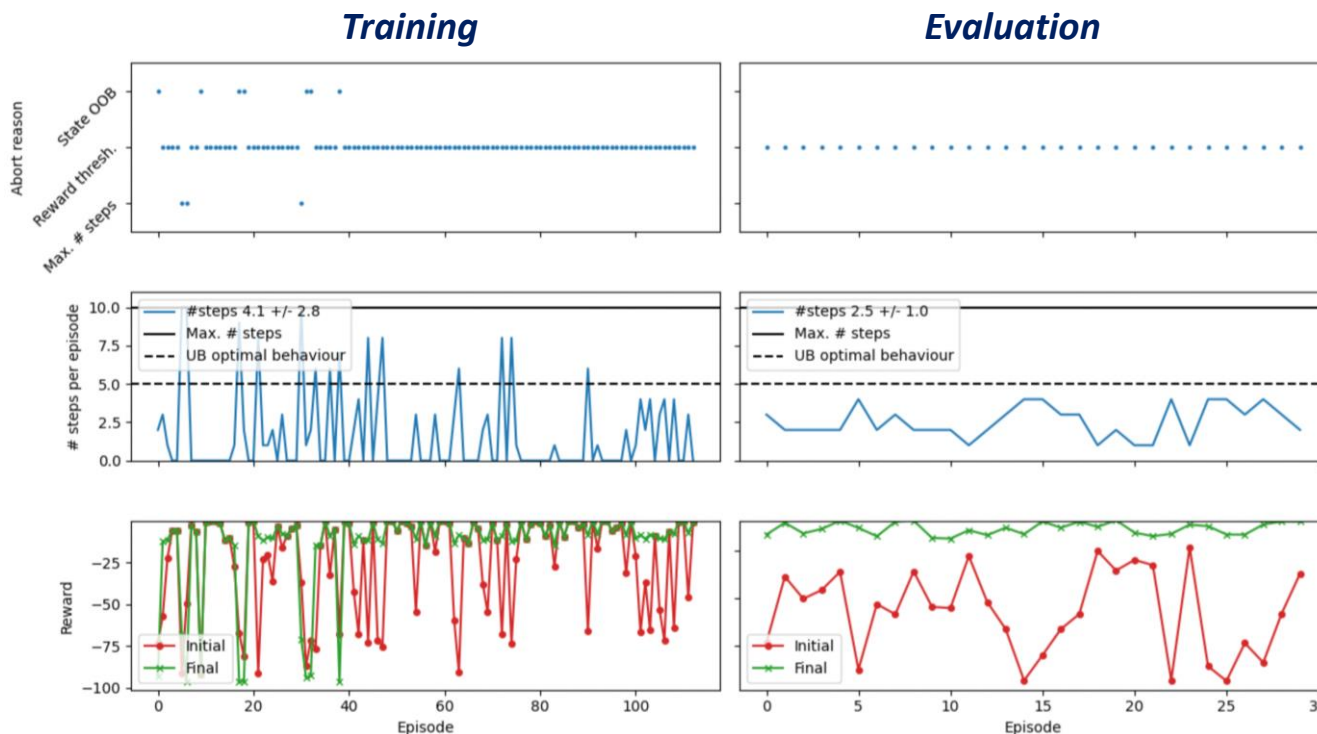
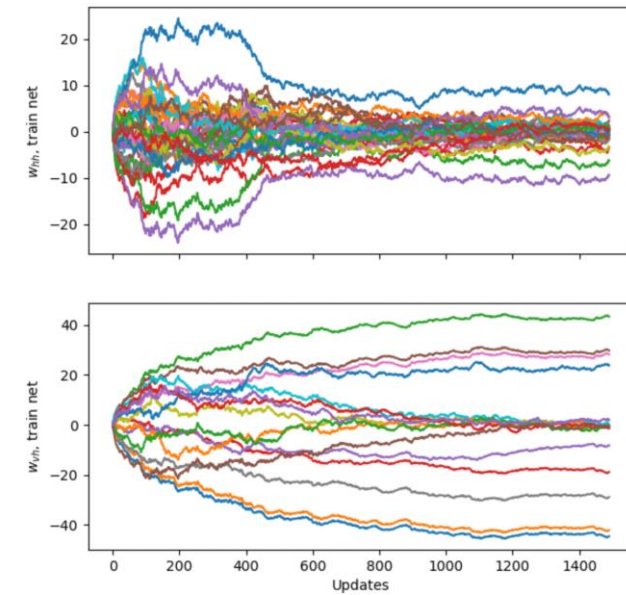


- QBM coupling weights trained with SQA and on QPU
 - Similar distribution and convergence time scales

Alternative approaches

RL with QAOA

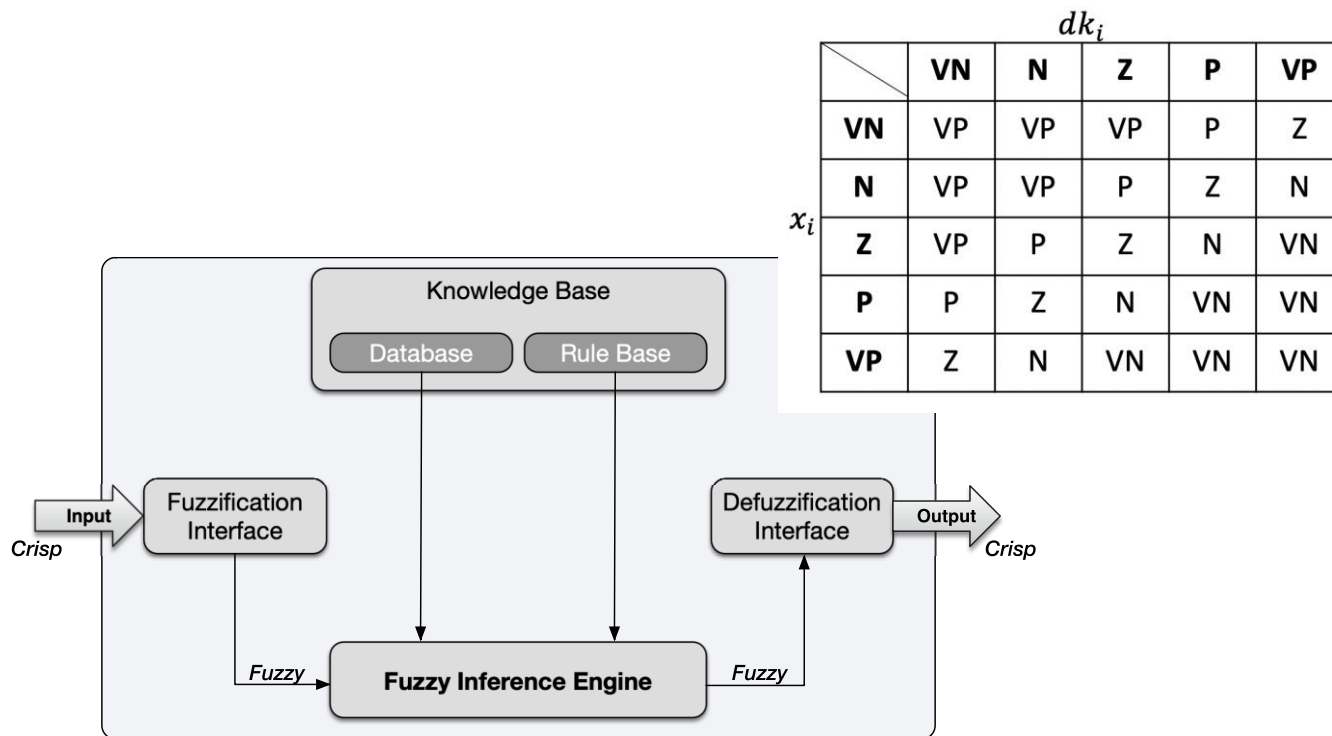
- [QAOA: Quantum Approximate Optimization Algorithm](#)
- **Solver** for combinatorial optimization problems: finds spin configuration with minimum energy; **not based on annealing, but more generic gate-based QPU**
- Can solve **quadratic unconstrained binary optimization (QUBO)** problems
- **Works well, but simulations compute-intensive (~5.5 h for 100 interactions)**
- On hardware (e.g. IBM), could be affected by noise



Alternative approaches: quantum fuzzy logic controller

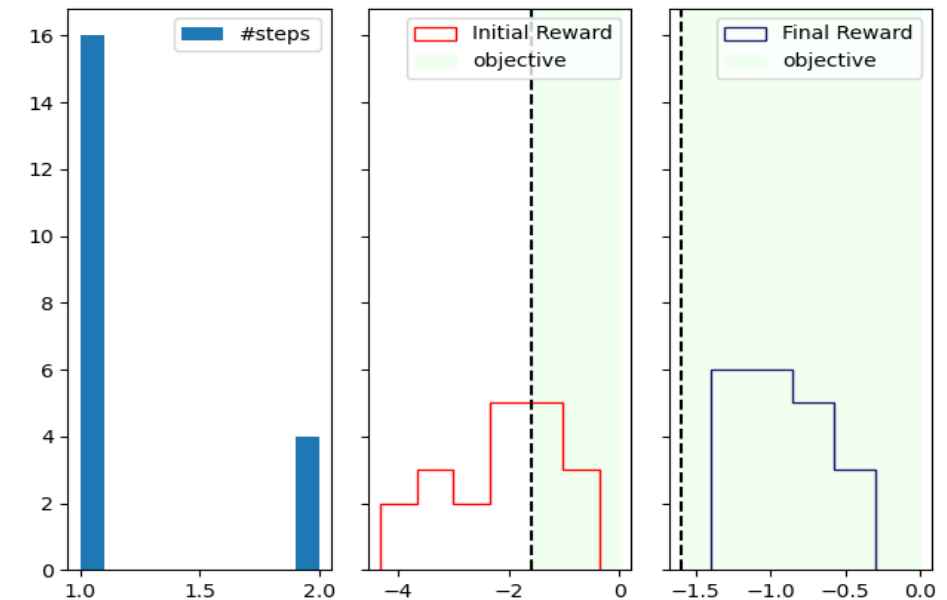
Evaluation on AWAKE beam line

- **Alternative control algorithm**
- **Fuzzy Logic** is used to develop control systems **based on linguistic rules** → **highly interpretable**
- **Quantum Fuzzy Control System** (*G. Acampora, R. Schiattarella, A. Vitiello*)
Exploit **exponential advantage** in computing fuzzy rules on quantum computers
- **Successfully evaluated on AWAKE beam line (no training required)**



Evaluation: on AWAKE beam line

Objective reached typically in 1 step



QUBO formulation

- **QUBO**: quadratic unconstrained binary optimization problems
- **The kind of problems that quantum computers (annealers) solve efficiently**
- **Example with 3 qubits**

$$f(x_1, x_2, x_3) = -3x_1 + x_1 x_2 - x_2 x_3$$

$$Q = \begin{bmatrix} -3 & 1 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}$$

- **QUBO matrix**
 - On diagonal terms describe self-couplings, i.e. biases
 - Off diagonal terms describe quadratic couplings between qubits

- **Run annealing 5000 times**

```
[1 0 0]: 33.4 % of occurrences  
[1 0 1]: 33.2 % of occurrences  
[1 1 1]: 33.4 % of occurrences
```

Corresponds to our solution table

```
# Solution table:  
# x1  x2  x3    f(x)  
# 0   0   0     0  
# 1   0   0    -3  (*)  
# 0   1   0     0  
# 0   0   1     0  
# 1   1   0    -2  
# 0   1   1    -1  
# 1   0   1    -3  (*)  
# 1   1   1    -3  (*)
```